Low-Complexity Interleaver Design for Turbo Codes

A Thesis
Presented to
The Academic Faculty

by

Nancy B. List

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
June 3, 2004

Low-Complexity Interleaver Design for Turbo Codes

Approved by:

Dr. Douglas Williams, Committee Chair

Dr. Faramarz Fekri

Dr. Steven McLaughlin

Date Approved: 3 June 2004

To my children,

my husband,

and my parents.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

A low-complexity method of interleaver design, sub-vector interleaving, for both parallel and serially concatenated convolutional codes (PCCCs and SCCCs, respectively) is presented here. Since the method is low-complexity, it is uniquely suitable for designing long interleavers.

Sub-vector interleaving is based on a dynamical system representation of the constituent encoders employed by PCCCs and SCCCs. Simultaneous trellis termination can be achieved with a single tail sequence using sub-vector interleaving for both PCCCs and SCCCs. In the case of PCCCs, the error floor can be lowered by sub-vector interleaving which allows for an increase in the weight of the free distance codeword and the elimination of the lowest weight codewords generated by weight-2 terminating input sequences that determine the error floor at low signal-to-noise ratios (SNRs). In the case of SCCCs, sub-vector interleaving lowers the error floor by increasing the weight of the free distance codewords. Interleaver gain can also be increased for SCCCs by interleaving the lowest weight codewords from the outer into non-terminating input sequences to the inner encoder.

Sub-vector constrained S-random interleaving, a method for incorporating S-random interleaving into sub-vector interleavers, is also proposed. Simulations show that short interleavers incorporating S-random interleaving into sub-vector interleavers perform as well as or better than those designed by the best and most complex methods for designing short interleavers. A method for randomly generating sub-vector constrained S-random interleavers that maximizes the spreading factor, S, is also examined.

The convergence of the turbo decoding algorithm to maximum-likelihood decisions

on the decoded input sequence is required to demonstrate the improvement in BER performance caused by the use of sub-vector interleavers. Convergence to maximum-likelihood decisions by the decoder do not always occur in the regions where it is feasible to generate the statistically significant numbers of error events required to approximate the BER performance for a particular coding scheme employing a sub-vector interleaver. Therefore, a technique for classifying error events by the mode of convergence of the decoder is used to illuminate the effect of the sub-vector interleaver at SNRs where it is possible to simulate the BER performance of the coding scheme.

CHAPTER I

INTRODUCTION

In 1948, Claude Shannon published his famous paper, "A Mathematical Theory of Communications," in which he determined that every communication channel has a maximum capacity for reliable transmission. Transmitting at a rate below capacity with a good code of sufficient block length, n, results in reliable communication. Transmitting at rates greater than channel capacity results in unreliable communication, no matter how good the code. Shannon's theory promised the existence of codes for which the bit error rate (BER) in the received sequence could be made arbitrarily small for increased block length, n. The goal of researchers for the past half-century has been to find these codes.

Turbo codes are a class of error correcting codes introduced in 1993 which came closer to approaching this theoretical limit than any other class of error correcting codes known at that time. Furthermore, they achieved their remarkable performance with relatively low complexity encoding and decoding algorithms.

Random interleaving is key to the remarkable performance of turbo coding schemes at low signal to noise ratios (SNRs). Previous attempts to optimize these interleavers fall into one of two categories: interleavers chosen by a random search performed to optimize a particular cost function as in [15],[17], [37], and [30], or specially designed linear block interleavers as in [4], [12], [14], and [26]. The best methods for designing short interleavers involve random searches, which make these approaches impractical for designing long interleavers. The simplest methods for optimizing interleavers are based on block interleavers and have resulted in designs that, while performing well for short interleavers, do not achieve the reduction in BER that is expected with

increased interleaver length.

The goal of this research is to develop low-complexity methods of designing interleavers for turbo codes, to explain analytically the improvements in BER performance caused by the use of these interleavers, to compare the performance of turbo codes employing these interleavers to the performance of turbo codes employing interleavers designed by other methods, and to verify the improvements in BER performance that were predicted analytically.

1.1 Overview

This dissertation is divided into three main parts:

- Background information on turbo codes in Chapter 2,
- Original contributions of the research performed in Chapters 3-6,
- Summary of the work done and possible directions for future research in Chapter 7.

Chapter 2 contains background information on turbo codes, including a description of the constituent encoders employed by turbo coding schemes, the difference between parallel and serially concatenated turbo coding schemes, a description of the algorithms used to decode turbo codes, and the purpose of the interleaver in a turbo coding scheme. Different approaches to interleaver design reported in the literature are discussed in this chapter. Methods of analyzing turbo codes that were used in this research are described in Chapter 2 as well.

In Chapter 3, the first original contributions of this research are reported. In this chapter, a method of modeling the constituent encoders in a turbo coding scheme as dynamical systems is described. The dynamical system model of the constituent encoders is key to understanding interleaver properties that led to the low-complexity method of interleaver design, "sub-vector interleaving," that is the main contribution

of this research. Chapter 3 addresses the low-complexity design of interleavers for parallel concatenated convolutional codes (PCCCs). It is shown in this chapter how sub-vector interleaving can allow for dual trellis termination with a single tail sequence appended to the input sequence to the encoder. The performance of a PCCC employing a sub-vector interleaver is predicted analytically, and then the performance is verified with simulations of the PCCC where the encoded sequences are transmitted over a channel corrupted by additive, white, Gaussian noise (AWGN) at different noise levels.

Sub-vector interleaving for serially concatenated convolutional codes (SCCCs) is presented in Chapter 4. This method of interleaving has the added benefit of increasing the interleaver gain of an SCCC. An upper bound on the interleaver gain for an SCCC, which is dependent on the constituent encoders employed by the SCCC, is derived in this chapter. As in Chapter 3, a method of modeling the constituent encoders as dynamical systems, which led to the method of sub-vector interleaving for SCCCs, is presented. Termination of the inner and outer encoders of the SCCC with a single tail sequence appended to the input sequence to the overall encoder is possible with sub-vector interleaving, a result not presented in the literature for any other method of interleaver design for SCCCs. The BER performance of SCCCs employing sub-vector interleavers is predicted analytically and is also verified by computer simulation.

A method of incorporating S-random interleaving [17] into sub-vector interleaving is outlined in Chapter 5. S-random interleaving is a simple, but effective, method of generating interleavers for turbo codes. It is widely used in simulations reported in the literature because of the simple nature of the algorithm used to design S-random interleavers, the effectiveness of the resulting interleaver in improving BER performance of the coding scheme, and the adaptability of the method to turbo coding schemes

employing different types of constituent encoders. In Chapter 5, it is shown that S-random interleaving incorporated into sub-vector interleaving (sub-vector constrained S-random interleaving) results in an improvement to both methods of interleaving. The S-parameter in an S-random interleaver describes the spreading of the bits in the input sequence that the interleaver causes. The maximum possible spread, S_{max}, is discussed in this chapter and a technique for generating S_{max}-random interleavers is outlined. The effect of the sub-vector constraint on S in a sub-vector constrained S-random interleaver is also discussed, and a method for randomly generating sub-vector constrained S_{max}-random interleavers is described. Simulations of the BER performance of PCCCs and SCCCs employing sub-vector constrained S-random interleavers are also presented in this chapter.

Analysis of turbo codes to quantify the improvement in bit-error rate (BER) performance caused by the use of sub-vector interleavers proved to be difficult, especially in the case of SCCCs. Sub-vector interleavers improve the performance of coding schemes by reducing the number of low weight codewords and increasing the weight of the free distance codewords for a particular PCCC or SCCC. The convergence of the turbo decoding algorithm to maximum-likelihood decisions on the decoded input sequence is required to demonstrate the improvement in BER performance caused by the use of sub-vector interleavers. Convergence to maximum-likelihood decisions by the decoder did not occur in the regions where it was feasible to generate the statistically significant numbers of error events required to approximate the BER performance of the coding schemes employing sub-vector interleavers in the case of SCCCs. Therefore, a technique for classifying error events by the mode of convergence of the decoder was used to illuminate the effect of the sub-vector interleaver at SNRs where it was possible to simulate the BER performance of the coding scheme. These results are presented in Chapter 6.

Chapter 7 is a summary of the original contributions and results presented in this dissertation. Suggestions are made for future areas of research in interleaver design for turbo codes that seem promising in light of these contributions.

CHAPTER II

BACKGROUND: TURBO CODES

Turbo codes comprise two or more convolutional encoders connected in parallel or in series by an interleaver. We refer to codes generated by encoders connected in parallel as "parallel concatenated convolutional codes," or PCCCs. We refer to codes generated by encoders connected in series as "serially concatenated convolutional codes," or SCCCs. The constituent encoders in a PCCC are always recursive convolutional encoders. An SCCC must have an inner recursive convolutional encoder, but the outer encoder may be either recursive or non-recursive.

2.1 Recursive Convolutional Encoders

A recursive convolutional encoder is a convolutional encoder with feedback. The encoder in Fig. 1 is a three delay state recursive convolutional encoder. In this example, the outputs from the last two delay states are fed back into the input. The parity bits for this encoder are a mod 2 sum of the input to the encoder and the output from each of the delay states.

Since the encoder has feedback, it generates an infinite weight parity sequence in response to an impulse (i.e., a single non-zero input bit). Input sequences that drive an encoder away from the zero state and then back to the zero state before the end of the sequence (possibly multiple times) are called "terminating input sequences." The lowest weight terminating input sequence possible for a recursive convolutional encoder has weight 2. This is in contrast to a non-recursive convolutional encoder for which weight 1 terminating input sequences exist.

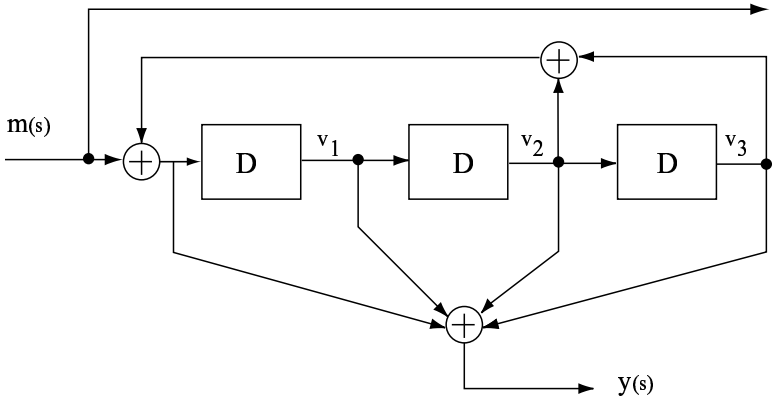

Figure 1: Recursive convolutional encoder with three delay states and overall rate $= \frac{1}{2}$

The generator matrix for the recursive convolutional encoder in Fig. 1 is given by

$$G = \left[1, \frac{n(D)}{d(D)}\right] = \left[1, \frac{1 + D + D^2 + D^3}{1 + D^2 + D^3}\right]$$

It is shown in [9] that a weight-2 input sequence of length $= 2^k$, where k is the number of delay states of the encoder and $2^k - 1$ is the period of the feedback polynomial of the encoder, $d(D)$, is the shortest weight-2 terminating input sequence for a recursive convolutional encoder. Thus, a length-8 input sequence

$$\underline{m} = \left[1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\right]$$

is a terminating input sequence for the recursive convolutional encoder in Fig. 1.

2.2 Parallel vs. Serial Concatenation

Parallel concatenation was the original configuration, proposed in the seminal paper published by Berrou, et al., in [12], of the constituent encoders and interleaver that comprise a turbo coding scheme. Though Berrou, et al., were able to demonstrate the superb performance of parallel concatenated turbo codes in their seminal paper, they did not initially offer an explanation of this performance. Numerous authors, including Berrou, et al., sought to explain the performance of parallel concatenated turbo codes and the techniques of designing and decoding parallel concatenated turbo codes in [6], [9], [11], [42], and [43], in addition to many others.

In the parallel concatenated turbo coding scheme shown in Fig. 2, the input sequence $\underline{m}$ is encoded by the top encoder. An interleaved version of the input sequence, $\underline{m}_\pi$, is encoded by the lower encoder. We assume that the decoder has knowledge of the interleaver, and thus the input sequence must only be transmitted once across the channel. So, the overall rate for the turbo coding scheme in Fig. 2 would be $\frac{1}{3}$, before puncturing.

Serially concatenated turbo coding and decoding was introduced by Benedetto, et al., in [8], and further developed in [10] and [7]. In a serially concatenated turbo

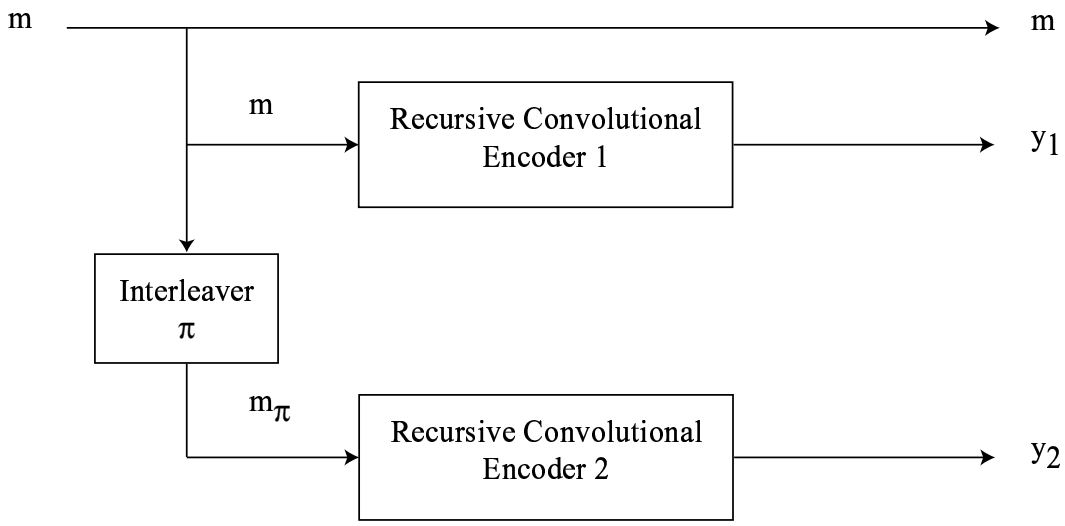

Figure 2: Parallel concatenated convolutional encoding scheme

coding scheme, the input sequence $\underline{m}$ is encoded only by the outer encoder. The parity bits from the outer encoder are interleaved into the sequence $\underline{y}_{1,\pi}$ and then encoded by the inner encoder. The inner encoder generates the sequence $\underline{y}_2$. Fig. 3 shows a flow diagram of a serially concatenated coding scheme.

2.3 Decoding Algorithm

In the parallel concatenated turbo coding scheme, the input sequence to each of the encoders are decoded separately using a soft-output a posteriori probability (APP) decoding algorithm. Information about the decoded input sequence is passed iteratively between the two decoders, until the decoders converge to a decision on the input sequence. The decoder output at the final step is given by $\widehat{m}_B$ in Fig. 4.

In the serially concatenated turbo coding scheme, the parity sequence output by the outer encoder and the input sequence to the inner encoder are decoded separately by the soft-output APP decoding algorithm. The input sequence to the outer encoder is decoded as a final step taken after the outer and inner encoders have converged to a decision on the interleaved sequence passed between them. The decoder output at

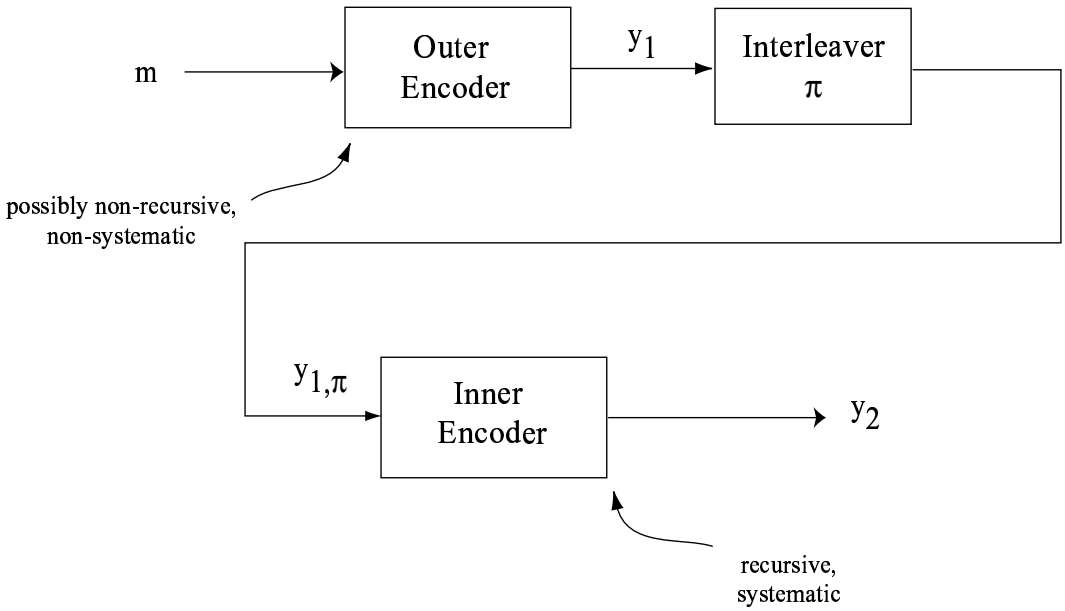

Figure 3: Serially concatenated convolutional encoding scheme

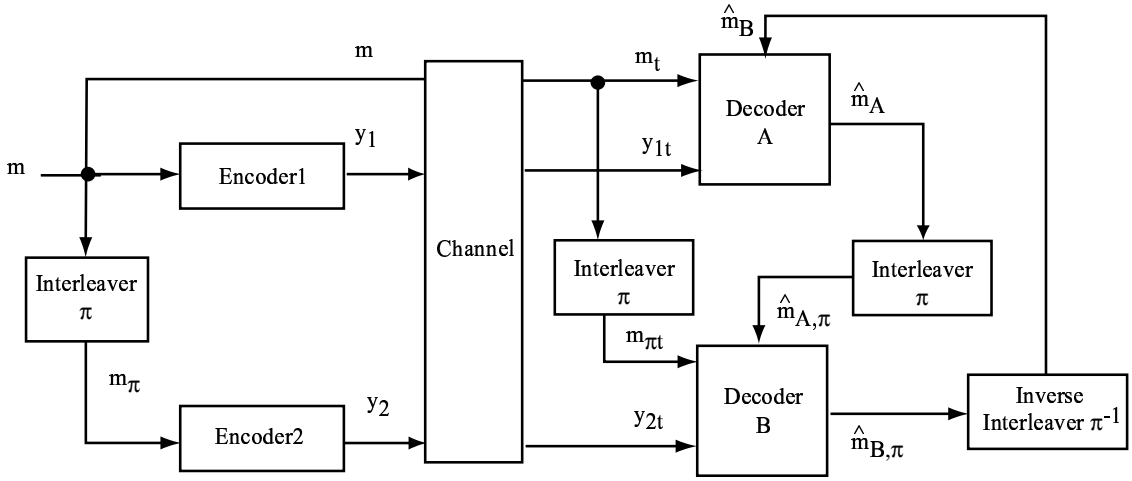

Figure 4: Flow diagram of the encoder, channel, and decoder for a parallel concatenated convolutional coding scheme

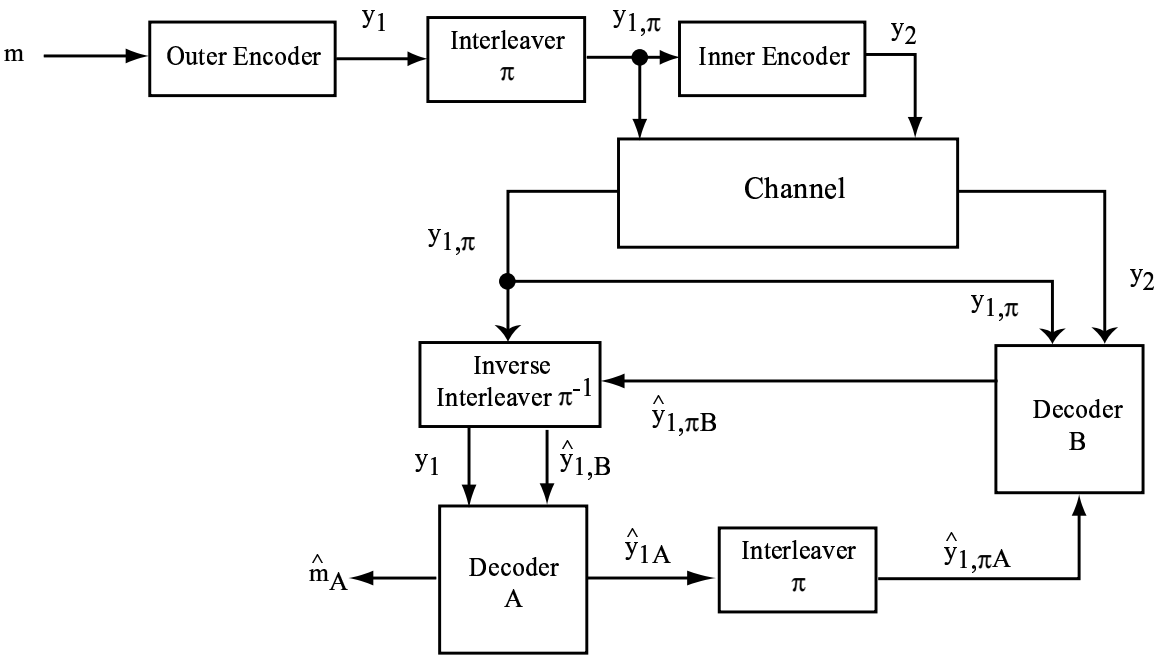

Figure 5: Flow diagram of the encoder, channel, and decoder for a serially concatenated convolutional coding scheme

the final step is given by $\widehat{m}_A$ in Fig. 5.

There are many variations on the decoding algorithm. The original turbo decoder was based on the Bahl, Cocke, Jelinek, Raviv (BCJR) algorithm [3], which was developed around the same time as the better known Viterbi algorithm, for decoding convolutional codes.

The BCJR algorithm seeks to maximize the probability that each bit in the input sequence is decoded correctly. This is different from the Viterbi algorithm, which seeks to maximize the likelihood that the entire code sequence is decoded correctly. The BCJR algorithm is roughly twice as complex as the Viterbi algorithm. The primary reason it is used in the turbo decoding algorithm is that its output is a soft decision on the input bits, which makes it suitable for use in an iterative decoder. It has also been shown to have superior performance to the Viterbi algorithm at low SNRs, albeit at a cost of twice the complexity of the Viterbi algorithm.

Less complex decoding algorithms for turbo codes also exist, such as the soft

output Viterbi algorithm (SOVA) proposed for use in decoding turbo codes in [24]. These algorithms are inferior to the BCJR algorithm for decoding turbo codes in terms of BER performance. Therefore, the research described in this proposal relies on the BCJR algorithm in its simulations.

2.4 Purpose of the Interleaver

The primary function of the interleaver is to improve the distance properties of the concatenated coding scheme. In PCCCs, the ideal interleaver permutes input sequences that generate low weight codewords from one encoder into input sequences that generate high weight codewords from the other encoder. In SCCCs, the ideal interleaver permutes low weight codewords from the outer encoder into input sequences generating high weight codewords from the inner encoder.

To a lesser extent the interleaver also serves to reduce the correlation between the input sequence and the parity bits associated with the interleaved input sequence. Because an independence assumption is made on the sequence being decoded and the extrinsic information related to the sequence, it is important to make sure that the input sequence and the parity bits associated with the interleaved input sequence are as uncorrelated as possible. It was shown in [39] that addressing this issue when designing interleavers improves the convergence properties of PCCCs and SCCCs employing short interleavers. Long interleavers (i.e., length-$N > 500$) selected randomly have been shown in [28] to have good correlation properties, as good as long interleavers designed specifically for those properties.

Low weight codewords in a turbo coding scheme are generated in a two step process. In the first step, an input sequence that begins with one of the constituent encoders in the all-zero state and returns that encoder to the all-zero state at the end of the input sequence is encoded. The parity sequence generated by such an input sequence terminates when the last non-zero bit of the input sequence is encoded. If

the distance between the first and last non-zero bit of this terminating input sequence is small, then the codeword it generates will have relatively low Hamming weight. In the second step, either the input sequence itself or the parity sequence it generates (in the case of a PCCC or SCCC, respectively) is interleaved so that another low weight codeword is generated by the other constituent encoder.

Example 1 *Assuming that the parallel concatenated coding scheme in Fig. 2 employs as its constituent encoders the recursive convolutional encoder shown in Fig. 1, the upper encoder in Fig. 2 is terminated by the length-14 input sequence*

$$\underline{m} = \left[\, 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \,\right],$$

and generates the parity sequence

$$\underline{y}_1 = \left[\, 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \,\right].$$

If $\underline{m}$ *is interleaved into*

$$\underline{m}_\pi = \left[\, 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \,\right],$$

the lower encoder in Fig. 2 is also terminated and generates the parity sequence

$$\underline{y}_2 = \left[\, 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \,\right].$$

The Hamming weight of the overall codeword generated in this case is the Hamming weight of $\underline{m} + \underline{y}_1 + \underline{y}_2 = 3 + 4 + 4 = 11$.

However, if $\underline{m}$ *is interleaved into*

$$\underline{m}_\pi = \left[\, 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0 \,\right],$$

the lower encoder in Fig. 2 is not terminated and generates the parity sequence

$$\underline{y}_2 = \left[\, 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1 \,\right].$$

In this case the Hamming weight of $\underline{m} + \underline{y}_1 + \underline{y}_2 = 3 + 4 + 7 = 14$. *Furthermore, the Hamming weight of the parity sequence* $\underline{y}_2$ *would increase with increased block length since the input sequence* $\underline{m}_\pi = \left[\, 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1 \,\right]$ *did not terminate the lower encoder.*

Because of the infinite impulse response of the constituent recursive convolutional encoders and the use of a random interleaver to couple the concatenated encoders, relatively few low weight codewords exist in a turbo coding scheme. This phenomena, described as "spectral thinning" in [36], is the cause of the very low BERs of turbo coding schemes at low SNRs.

2.4.1 Spectral Thinning and Random Interleavers

PCCCs always employ recursive convolutional encoders, which have an infinite impulse response, as the constituent encoders. Therefore, most input sequences to PCCCs generate parity sequences that are not terminated (i.e., the parity sequence is terminated artificially at the end of the input sequence, but would not terminate given an infinite length input sequence) in one or both of the encoders. We define a terminating input sequence to be an input sequence that drives both of the constituent encoders back to the zero state before the end of the sequence.

The situation is slightly different in the case of SCCCs. The outer encoder in a SCCC typically is not a recursive convolutional encoder, but some other type of encoder with a high weight free distance codeword. Employing a non-recursive outer encoder with a high weight free distance codeword in an SCCC improves the overall distance properties of the SCCC by increasing the spectral thinning caused by the interleaver, a topic that will be addressed in the following sections.

2.4.1.1 Parallel Concatenated Convolutional Codes

A random interleaver used in a PCCC achieves spectral thinning for the overall code by permuting low weight codewords in one encoder into similar low weight codewords in the second encoder with a probability proportional to $\frac{1}{N^{d-1}}$, where N is the length of the interleaver and d is the weight of the input sequence generating the codeword. We see the effect of spectral thinning when we calculate the probability of a bit error resulting from an error pattern that corresponds to a single terminated codeword,

where error patterns are caused by the channel corrupting the signal with additive white Gaussian noise. A single terminated codeword is a codeword generated by a weight-d terminating input sequence that causes both of the constituent encoders to diverge from and reemerge to the zero state exactly once. If p_1 and p_2 are the parity sequences from the constituent recursive convolutional encoders generated by the weight-d terminating input sequence, then an input weight-d terminated error pattern, $\underline{e}_{d,p}$, causes d bit errors when it is incorrectly decoded as a terminated codeword with Hamming weight $p = d + p_1 + p_2$. Using the idea of a uniform random interleaver developed in [9], which performs as the average of all length-N random interleavers, the probability of a bit error resulting from a terminated error pattern $\underline{e}_{d,p}$ incorrectly decoded as a single terminated codeword can be expressed as:

$$\begin{aligned} P_b\left(e \mid \underline{e}_{d,p}\right) &\approx \left(\tfrac{d}{N}\right) \frac{(N-l_1+1)(N-l_2+1)}{\binom{N}{d}} Q\left(\sqrt{p \cdot \tfrac{2R_c\varepsilon_b}{N_0}}\right) \\ &< \left(\tfrac{d}{N}\right) \cdot (d!) \cdot \left(N^{2-d}\right) \cdot e^{-p\left(\frac{R_c\varepsilon_b}{N_0}\right)}, \end{aligned} \tag{1}$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{\frac{-t^2}{2}} dt$, R_c is the overall rate of the turbo coding scheme, $\frac{\varepsilon_b}{N_0}$ is the SNR per bit, and l_1 and l_2 are the distances between the first and last non-zero bits of the input sequence of the terminated codeword before and after interleaving, respectively.

We define an input weight-d multiply terminated codeword as a codeword generated by a weight-d terminating input sequence that drives the constituent encoders to and from the zero state multiple times. The probability of a bit error resulting from an error pattern $\underline{e}_{d,p}$ decoded as an input weight-d multiply terminated codeword where the parity sequence intersects the zero state in the first encoder n_1 times and in the second encoder n_2 times is calculated as

$$P_b\left(e \mid \underline{e}_{d,p}\right) \approx \left(\frac{d}{N}\right) \frac{\binom{N-l_1+n_1}{n_1}\binom{N-l_2+n_2}{n_2}}{\binom{N}{d}} Q\left(\sqrt{p \cdot \frac{2R_c\varepsilon_b}{N_0}}\right)$$
$$< \left(\frac{d}{N}\right) \cdot \left(\frac{d!}{n_1!n_2!}\right) \cdot \left(N^{n_1+n_2-d}\right) \cdot e^{-p\left(\frac{R_c\varepsilon_b}{N_0}\right)},$$

where l_1 and l_2 are the sum of the distances between the first and last non-zero bits of the n_1 and n_2 individual terminated codewords, respectively [36].

Since the constituent encoders in a turbo coding scheme are recursive, there are no weight-1 input sequences that generate terminated codewords. We see in (1) that the probability of a bit error resulting from a single weight-d input sequence that generates a terminated parity sequence decreases at a rate proportional to $\frac{1}{N^{d-1}}$. Therefore, input weight-2 single terminated error patterns contribute to the BER with probability inversely proportional to N, while all single terminated error patterns with higher input sequence weight contribute to the BER with probabilities inversely proportional to higher powers of N.

Another class of error patterns with bit error probability inversely proportional to N correspond to codewords with an even input weight d that generate parity sequences that diverge from and reemerge to the zero state the maximum number of times possible. The maximum number of times the parity sequence can diverge and reemerge with the zero state is given by $r = \left\lfloor \frac{d}{2} \right\rfloor$. Terminated error patterns corresponding to these types of codewords can be described as the concatenation of individual input weight-2 terminated error patterns. The probability of a bit error resulting from this type of error pattern is calculated as

$$P_b\left(e \mid \underline{e}_{d,p}\right) \approx \left(\frac{d}{N}\right) \frac{\binom{N-l_1+\frac{d}{2}}{\frac{d}{2}}\binom{N-l_2+\frac{d}{2}}{\frac{d}{2}}}{\binom{N}{d}} Q\left(\sqrt{p \cdot \frac{2R_c\varepsilon_b}{N_0}}\right)$$
$$< \left(\frac{d}{N}\right) \cdot \left(\frac{d!}{\frac{d}{2}!\frac{d}{2}!}\right) \cdot e^{-p\left(\frac{R_c\varepsilon_b}{N_0}\right)}.$$

As in the case of single input weight-2 terminated error patterns, these terminated error patterns have bit error probabilities proportional to $\left(\frac{1}{N}\right)$. However, they are associated with at least twice the number of parity bits as the single input weight-2 terminated error patterns and, therefore, do not contribute as significantly to the overall BER of a turbo coding scheme.

2.4.1.2 Serially Concatenated Convolutional Codes

As for PCCCs, a random interleaver used in an SCCC achieves spectral thinning for the overall code by permuting low weight codewords from the outer encoder into input sequences resulting in low-weight parity sequences from the inner encoder with a probability inversely proportional to powers of the interleaver length, N_{in}.

Because the input sequences to the inner encoder are the output codewords from the outer encoder in an SCCC, the minimum weight of the input sequence to the inner recursive convolutional encoder in an SCCC is typically greater than two. The spectral thinning that occurs in an SCCS is, therefore, greater than what occurs in a PCCC. We see this when we calculate the probability of a bit error resulting from an error pattern $\underline{e}_{d_{free},p}$, which corresponds to the free distance weight-d_{free} codeword of the outer encoder, that is interleaved into an input sequence resulting in a weight-p parity sequence from the inner encoder. Using the idea of a uniform random interleaver, as before, we can calculate the probability of a bit error resulting from a particular error pattern $\underline{e}_{d_{free},p}$. Assuming that this error pattern corresponds to a weight-d_{free} input sequence to the inner encoder that causes that encoder to diverge and reemerge from the zero state *exactly once*, the probability of a bit error resulting from $\underline{e}_{d_{free},p}$ is calculated as:

$$P_b\left(e|\underline{e}_{d_{free},p}\right) \approx \left(\frac{w}{N_{out}}\right) \frac{\left(N_{d_{free}}^{out}\right)(N_{in}-l+1)}{\binom{N_{in}}{d_{free}}} Q\left(\sqrt{(d_{free}+p)\cdot\frac{2R_c\varepsilon_b}{N_0}}\right)$$
$$\propto (N_{in})^{1-d_{free}} \cdot e^{-\left(d_{free}+p\right)\left(\frac{R_c\varepsilon_b}{N_0}\right)},$$

where w is the weight of the input sequence to the outer encoder, N_{out} is the length of the input sequence to the outer encoder, N_{in} is the length of the input sequence to the inner encoder, $N_{d_{free}}^{out}$ is the number of weight-d_{free} codewords from the outer encoder, l is the distance from the first to last non-zero bit of the interleaved error pattern, $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{\frac{-t^2}{2}} dt$, R_c is the overall rate of the SCCC, and $\frac{\varepsilon_b}{N_0}$ is the SNR per bit.

Low weight codewords from the outer encoder that are interleaved into codewords that cause the inner encoder to diverge and reemerge with the zero state more than one time correspond to another set of important error patterns in an SCCC. Since the inner encoder in an SCCC is recursive, there are no weight-1 input sequences to the inner encoder that cause it to diverge and reemerge with the zero state. At a minimum, a weight-2 input sequence is required for this to happen. If $d_{free} > 3$, it is possible that an error pattern corresponding to a codeword for the outer encoder could be interleaved into multiple concatenated terminated error patterns for the inner encoder. If d_{free} is even, such an error pattern may comprise a maximum of $\left(\frac{d_{free}}{2}\right)$ terminated error patterns for the inner encoder; if d_{free} is odd, it may comprise a maximum of $\left(\frac{d_{free}-1}{2}\right)$ terminated error patterns.

The probability of a bit error resulting from an error pattern corresponding to an odd weight-d_{free} codeword for the outer encoder that is interleaved so that it corresponds to the concatenation of the maximum number of terminated error patterns for the inner encoder is calculated as

$$\begin{aligned} P_b\left(e|\underline{e}_{d_{free},p}\right) &\approx \left(\frac{w}{N_{out}}\right) \frac{\left(N_{d_{free}}^{out}\right)\begin{pmatrix} N_{in}-l+\frac{\left(d_{free}-1\right)}{2} \\ \frac{\left(d_{free}-1\right)}{2} \end{pmatrix}}{\begin{pmatrix} N_{in} \\ d_{free} \end{pmatrix}} Q\left(\sqrt{(d_{free}+p)\cdot\frac{2R_c\varepsilon_b}{N_0}}\right) \\ &\propto (N_{in})^{-\frac{\left(d_{free}+1\right)}{2}} \cdot e^{-\left(d_{free}+p\right)\left(\frac{R_c\varepsilon_b}{N_0}\right)}. \end{aligned} \tag{2}$$

Similarly, the probability of a bit error resulting from an error pattern corresponding to an even weight-$(d_{free}+1)$ codeword for the outer encoder that is interleaved so

that it corresponds to the concatenation of the maximum number of terminated error patterns for the inner encoder is proportional to

$$P_b\left(e|\underline{e}_{d_{free}+1,p}\right) \propto (N_{in})^{-\frac{\left(d_{free}+1\right)}{2}} \cdot e^{-\left(d_{free}+1+p\right)\left(\frac{R_c\varepsilon_b}{N_0}\right)}. \tag{3}$$

We see from (2) and (3) that spectral thinning in SCCCs is inversely proportional to higher powers of the interleaver length, N_{in}, than for PCCCs. However, the reduction in BER resulting from spectral thinning calculated in (2) and (3) assumes maximum likelihood decoding is possible. Since the turbo decoding algorithm does not converge to the maximum likelihood decisions at low and moderate SNRs in the "waterfall" region of the BER performance curve, maximum likelihood decoding is not available at the SNRs of interest in some systems. Also, PCCCs have been shown in [10] to have better convergence properties than SCCCs in the waterfall region. Therefore, PCCCs can outperform SCCCs of the same complexity and decoding delay at low and moderate SNRs.

2.5 Approaches to Interleaver Design

Since the interleaver is the key to the BER performance of PCCCs and SCCCs at low SNRs, much attention has been paid to the design of interleavers to be used in these applications. The approaches to designing interleavers that improve the BER performance of a PCCC or SCCC range from heuristic searches, suitable for only short interleavers because of their complexity, to simple block interleavers that fall short of ideal, especially for designing long interleavers. Two of the approaches fall somewhere in between: a technique based on a fairly simple semi-random search making it useful for designing short or long interleavers, and another approach that is the subject of this thesis's research. The following sections in this chapter will serve to describe the relative strengths and weaknesses of the known approaches to interleaver design for PCCCs and SCCCs.

2.5.1 Iterative interleaver growth algorithms

Iterative interleaver growth algorithms constitute the class of interleaver design methods that are most successful in improving the BER performance of PCCCs and SCCCs. These algorithms are initialized with a short interleaver and then grown to a desired length element by element while satisfying particular design criteria or minimizing a particular cost function.

2.5.1.1 S-random algorithm

The S-random interleaver originally was developed in [17] for use with PCCCs, but has since shown its suitability for use with SCCCs as well. The S-random algorithm generates permutations of any length, long or short, that perform well for use with either PCCCs or SCCCs. The idea behind the S-random algorithm is to permute elements in the input sequence that are separated by a distance less than S to positions in the output sequence that are separated by a distance greater than S.

The permutations are generated as follows: Select a randomly generated integer as the first element of the permutation. For each subsequent element of the permutation, randomly generate another integer and compare it to the S previously selected elements. If the current selected element of the permutation is within a distance $\pm S$ of any of the S previously selected elements, discard it and randomly select another integer. Repeat this process till the permutation is complete with every element satisfying the S-random criteria.

The S-random algorithm has two very attractive features:

- It is simple to implement and computationally efficient.
- It performs very well, considerably better than an average random interleaver does, in both PCCCs and SCCCs.

The drawbacks to the S-random algorithm include:

- Searching time increases with the desired amount of separation, S, and the length of the interleaver.

- The algorithm is not guaranteed to finish successfully in any case. (In practice, most experimenters choose $S < \sqrt{N/2}$ in order to achieve a solution in a reasonable amount of time.)

- The S-random algorithm is generic in the sense that it does not consider the constituent encoders of the PCCC or SCCC and the codewords that are most significant in terms of BER contribution in a particular system.

- Interleaving the concatenation of multiple codewords is not addressed by the algorithm.

The third drawback of the S-random algorithm is addressed by Feng, et al., in [19] and [44]. Their method of interleaver design adds another criterion to the S-random search routine. In addition to satisfying the spread requirement, their algorithm also analyzes the distance spectrum of the component codes and makes sure that their interleaver breaks up input patterns that are most significant in terms of BER contribution. This additional criterion, however, increases the searching time of the algorithm and is less likely to converge to a solution than the S-random algorithm.

The fourth drawback of the S-random algorithm is addressed by Fragouli, et al., in [21]. Their method attempts to consider spreading the concatenation of two or more low weight codewords into higher weight codewords. However, when double concatenations are considered, the complexity of the algorithm is $O(N^2)$, vs. $O(N)$ for the standard S-random algorithm.

2.5.1.2 Iterative interleaver growth algorithm of polynomial complexity

An iterative interleaver growth algorithm of polynomial complexity was presented by Daneshgaran and Mondin [15]. This method of interleaver design was tailored to a

particular coding scheme by considering the types of error patterns that are significant for the constituent encoders the coding scheme employs.

Daneshgaran and Mondin's technique for generating the interleaver permutation is unique and fundamental to their design method. They describe their interleavers as sliding-window transposition boxes. As the sliding window moves across the data, the data element at the far edge of the sliding window is either output from the sliding window or transposed with another data element within the sliding window. The length of the sliding window is equal to the delay of the permutation, or the maximum time between an element entering and exiting the sliding-window transposition box.

Daneshgaran and Mondin propose the design of interleavers that minimize a cost-function based on the types of error patterns determined to be significant to a particular coding scheme. Their interleavers grow iteratively to the desired length, with the cost function minimized at each step of the process.

The complexity of this design method depends on the types of errors on which the cost-function is based. Single terminated errors are most significant for short interleaver lengths. Considering only these types of error patterns, the complexity of the interleaver design is $O(N^3)$. If even a single double terminated error pattern is considered, however, the complexity increases to $O(N^4)$. They have determined that, for interleavers of length 120-500, double terminated error patterns contribute significantly to the BER. For interleavers longer than 500, triple and quadruple terminated error patterns are significant. Accounting for these error patterns increases the complexity of the interleaver design to $O(N^5)$ and $O(N^6)$, respectively.

2.5.2 Interleavers with Structure

Interleavers with structure range from the simplest rectangular interleaver (write by row, read by column), to more sophisticated constructions that take into account the code structure and the elimination of significant error patterns.

2.5.2.1 Block interleavers

Berrou, et al., proposed a rectangular interleaver for use in PCCCs in [11]. The method of interleaving proposed there was simple: write the sequence to be interleaved into a 120×120 matrix row by row, and read the interleaved sequence column by column. The method suffered, in particular, from error patterns at the very end of the code block, since these patterns were not broken with this type of interleaving, and from a high multiplicity of weight-4 terminating input sequences. In [36] it was shown that the multiplicity of the weight-4 terminating input sequences was a consequence of the rectangular interleaver itself, and that the multiplicity of these error patterns did not increase with interleaver length (i.e., no interleaver gain for these error patterns).

Fig. 6 illustrates the problem of weight-4 terminating input sequences in a block interleaver for PCCCs. In this example, we assume the interleaver is designed for a PCCC employing the three delay state encoders in Fig. 1 as constituent encoders. In Section 2.1, we explained that weight-2 input sequences of length $= 2^k$, where k is the number of delay states for a recursive convolutional encoder, are terminating input sequences for the encoder. Since the constituent encoders have three delay states, a length-8 input sequence to the upper encoder

$$\underline{m} = \left[1\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \right]$$

that is interleaved into an identical input sequence

$$\underline{m}_\pi = \left[1\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \right]$$

for the lower encoder generates terminated parity sequences from both constituent encoders. Fig. 6 illustrates how this happens for a length-70 block interleaver used with the PCCC in this example.

Perez, et al., point out in [36] that the number of weight-4 terminating input sequences in the example in Fig. 6 increases with the length of the interleaver. This

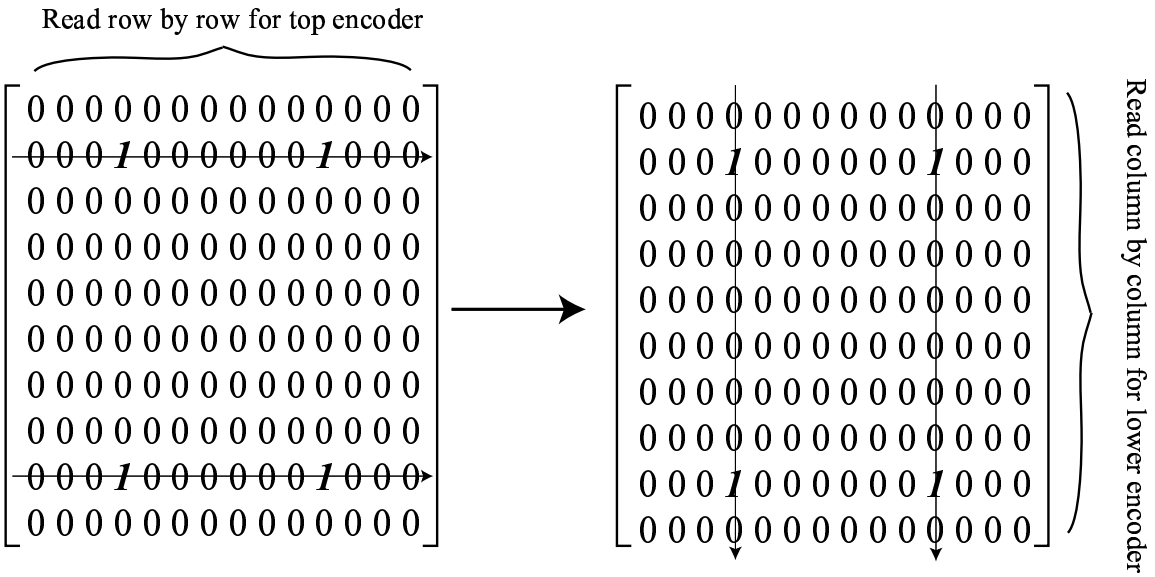

Figure 6: Weight-4 terminating input sequences that are not interleaved in a block interleaver for a PCCC employing the three delay state encoders in Fig. 1 as constituent encoders.

means that the interleaver gain is eliminated since the probability of a bit error due to an error pattern corresponding to one of these weight-4 terminating input sequences is no longer a function of the interleaver length:

$$P_b\left(e \mid \underline{e}_{4,p}\right) \approx C \cdot Q\left(\sqrt{p \cdot \frac{2R_c\varepsilon_b}{N_0}}\right),$$

where $C \approx$ a constant for all interleaver lengths. Thus, this error probability forms a floor on the BER performance of the PCCC that does not decrease with increased interleaver length.

An improvement on the block interleaver that addresses error patterns at the end of the block was proposed in [26]. This rectangular interleaver specifies that the input sequence be written row by row, and read column by column in the *reverse direction* that the input sequence was written. Reading in the reverse direction solves the problem of error patterns at the end of the input sequence, since error patterns at the end of the input sequence to one encoder are interleaved to the beginning of the input sequence for the other encoder.

2.5.2.2 Optimal period interleavers

Qi, et al., [37] propose designing interleavers for parallel concatenated turbo codes based on an optimal period interleaver. This method achieves some measure of weight spectrum thinning in the resultant turbo codes and an improvement in the BER performance over random interleaving.

An optimal period interleaver is one that permutes all terminated codewords with length equal to the period of the impulse response of the encoders (except for the all-zeros and all-ones input sequences) into non-terminating input sequences. The procedure for designing an interleaver based on an optimal period interleaver is as follows:

1. Perform a column permutation of a matrix with the input sequence bits read in row-wise by permuting the elements in each row of the matrix according to the optimal period interleaver.

2. Interleave along the columns of the matrix in a prescribed manner such that nearby elements of the matrix are separated far apart after interleaving.

Permuting the columns in the first step will break up all terminated codewords with length less than the period of the impulse response of the encoder (except for the all-zeros and all-ones codewords) into non-terminated codewords. Because weight-2 input sequences corresponding to terminated codewords are longer than one period of the impulse response of the encoders, column interleaving is necessary to increase the Hamming weight of the parity sequences associated with these problematic input sequences. Qi, et al., [37] used a method of interleaving described in [**?**] to perform column interleaving. They use the same interleaver for each column and then incrementally shift each column.

2.5.2.3 *Design of interleavers using the Hungarian method*

It is not possible to interleave all weight-2 input sequences corresponding to terminated codewords into non-terminating sequences. It is shown in [30] that at most it is possible to break all terminating weight-2 input sequences with $span < K^2$ where $K = 2^r - 1$ into non-terminating input sequences, leaving $1/K$ of the total number of terminating weight-2 input sequences unbroken.

A simple method for achieving this type of interleaving is to fill a $K \times N_{out}/K$ matrix with the elements of a length-N_{out} input sequence read in row-wise, and then circularly shift the elements of i^{th} row by i elements. Khandani proposes this type of interleaving in [30], but notes that it is not sufficient to break up the terminating weight-2 input sequences without taking into account weight-3 and higher input sequences. Khandani notes that his shift interleaver retains the property of breaking the weight-2 input sequences when interleaving is performed over the columns of the matrix. The method of interleaving he chooses to perform along the columns of the $K \times N_{out}/K$ matrix is based on the Hungarian method used to optimize the distance between elements in each column of the matrix. He does not consider interactions between elements in different columns in his method.

2.5.3 Design of Self-Terminated Interleavers

Hokfelt, et al., [27] outline the benefits of terminating at least one of the constituent encoders in a PCCC employing a pseudo-random interleaver. Appending a tail sequence to the input of one of the constituent encoders that drives the encoder back to the zero state can do this. Not terminating either of the constituent encoders increases the probability that bits (in a randomly or pseudo-randomly generated interleaver) at the end of the input sequence will be decoded incorrectly.

A method of designing interleavers for parallel concatenated turbo codes that leaves both constituent encoders in the same final state is proposed in [25] and [5].

By appending a single tail to the input sequence of the encoders, both encoders can be driven to the zero state at the end of the input sequence as long as the length-N interleaver satisfies a simple constraint:

$$\pi(i) \bmod L = i \bmod L,\ 0 \leq i \leq N-1$$

where L is the period of the constituent encoders of the parallel concatenated coding scheme.

Terminating the input sequences to the encoders improves the performance of the decoder at the end of the input sequence. Using a single tail appended to the input sequence to terminate both encoders as suggested in [25] and [5] also improves the throughput of the code by eliminating the need for a second tail sequence.

2.6 Turbo Code Analysis

The BER rate performance of PCCCs and SCCCs is determined by the free distance of the coding scheme and its interleaver, and by the convergence of the decoding algorithm to the bounds predicted by the code's distance properties. Section 2.6.1 outlines an efficient method used in this research for computing the free distance of a PCCC or SCCC. Section 2.6.2 discusses the modes of convergence of the iterative decoder used to decode SCCCs and PCCCs, and methods for predicting the SNRs where convergence of the decoder to maximum likelihood decisions on the decoded bits will occur.

2.6.1 Free Distance Computation

It is well known that the slope of the BER of a PCCC or SCCC is very steep at low and medium SNRs (the "waterfall" region of the error performance curves) but flattens out dramatically once the BER has converged to the error floor. The error floor is caused by error patterns corresponding to low weight codewords not being corrected by the decoder. The expected number, or multiplicity, of low weight codewords for a

particular turbo coding scheme employing a randomly generated interleaver decreases at a rate inversely proportional to the length of the interleaver in the case of a PCCC, or the length of the interleaver to powers ≥ 2 for a SCCC, as seen in Section 2.4.1. Thus, simply increasing the length of the randomly generated interleaver can lower the error floor.

Designing interleavers that can lower the error floor of a turbo code for a set length interleaver could improve the BER performance of the coding scheme. It would also allow the use of a shorter interleaver to achieve a given BER, which would reduce the latency in the transmitted data.

In order to evaluate the design of interleavers with good distance properties (i.e., higher free distance codewords with lower multiplicities), it is useful to be able to compute the actual distance properties of a particular interleaver. A fast algorithm for computing the distance properties of interleavers is described in [22].

The free distance codeword of a parallel concatenated turbo code can be computed using the following algorithm:[1]

1. Consider a set of length-i input sequences, $\overline{u}_i$, to the turbo encoder, where $0 < i \leq N$ and N is the length of the interleaver. Compute the Hamming weight of the parity sequence output by each of the encoders subject to the following conditions:

   (a) Assume that the free distance of the turbo code will be less than some nominal value, d^*.

   (b) For the encoder receiving the un-interleaved input sequence, compute the first i terms of the parity sequence, $\overline{v}_1(\overline{u}_i)$, and calculate its Hamming weight, $w(\overline{v}_1(\overline{u}_i))$.

[1] This is intended to be a broad description of the overall algorithm and, as such, a few details are omitted.

(c) For the encoder receiving the interleaved input sequence, compute the minimum weight path, $\overline{v}_2(\overline{u}_i)$, through the entire code trellis of the recursive convolutional encoder, constraining it to pass through edges of the trellis corresponding to the interleaved bits of the length-i input sequence. Garello, et al., [22] refer to this constrained minimum distance path through the trellis as the "constrained subcode." The Hamming weight of the constrained subcode is designated by $w(\overline{v}_2(\overline{u}_i))$.

2. If the Hamming weight of the input sequence, $w(\overline{u}_i)$, plus the Hamming weight of the parity sequences, $w(\overline{v}_1(\overline{u}_i)) + w(v_2(\overline{u}_i))$, is less than d^*, then set $d^* = w(\overline{u}_i) + w(\overline{v}_1(\overline{u}_i)) + w(v_2(\overline{u}_i))$.

3. All elements of $\overline{u}_i$ satisfying $w(\overline{u}_i) + w(\overline{v}_1(\overline{u}_i)) + w(v_2(\overline{u}_i)) \leq d^*$ survive to the next iteration. To create the set of elements $\overline{u}_{i+1}$, we take the surviving elements from $\overline{u}_i$ and append either a 0 or a 1 to the end of the input sequence to create two elements of $\overline{u}_{i+1}$ from each surviving element of $\overline{u}_i$.

4. Repeat from step 1 until $i = N$. At the end of the $i = N^{th}$ iteration, we have computed the free distance of the turbo code, $d_{free} = d^*$.

With a few simple modifications, this algorithm can be used to compute the multiplicity of the free distance codewords, as well as the multiplicities of higher weight codewords for use in a distance spectrum analysis of a turbo code and interleaver combination.

2.6.2 Convergence of Decoding Algorithm

The turbo code decoding algorithm is described as a discrete dynamical system in [1], iterating on the extrinsic information output from the constituent decoders. As a result, modes of convergence of the decoding algorithm can be described in terms of its properties as a dynamical system.

Agrawal and Vardy [1] offered proofs that in the extreme cases of the SNR approaching zero or infinity asymptotically, the turbo decoding algorithm has unique fixed points corresponding to mostly incorrect or correct decisions on the input sequence, respectively. In between these extreme SNRs (i.e., at practical SNRs), the turbo decoding algorithm exhibits three modes of convergence: convergence to an unequivocal fixed point, convergence to an indecisive fixed point, convergence to an invariant set. Simulations in [1] and [38] have demonstrated these types of convergence and the types of errors with which they are associated:

1. For the case of convergence to an unequivocal fixed point, most decisions by the decoder correspond to maximum-likelihood decoding of the received sequence. Errors that occur when the decoder has converged to an unequivocal fixed point typically correspond to low weight codewords for the turbo coding scheme.

2. In the case of convergence to indecisive fixed points, the extrinsic information passed between the two decoders remains very low, indicating that the decoding algorithm is ambiguous regarding the values of the information bits. These unequivocal fixed points can correspond to a large number of bit errors in the decoded sequence.

3. The case of convergence to an invariant set generally occurs in the "waterfall region" of the range of SNRs and occurs when an indecisive fixed point bifurcates. In practice this is observed as the decoder converging on a set of fixed points and oscillating among them periodically.

CHAPTER III

INTERLEAVER DESIGN FOR PARALLEL CONCATENATED CONVOLUTIONAL CODES

A typical PCCC, as shown in Fig. 7, consists of two systematic recursive convolutional encoders and an interleaver operating on the input bits to the second encoder. The input bits to the two encoders are the same, except that they are interleaved before entering the second encoder.

The discussion in this chapter is based on the PCCC shown in Fig. 7 consisting of two identical, rate $\frac{1}{2}$, eight state (or three delay state) recursive convolutional encoders with generator matrix $G = \left[1, \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$. However, it is straightforward to apply these ideas to PCCCs employing non-identical constituent encoders or PCCCs employing constituent encoders with fewer or more delay states than the encoder in Fig. 7. Chapter 4 will address the application of these ideas to SCCCs.

3.1 Analysis of Recursive Convolutional Encoders

This section presents a new, novel representation of a recursive convolutional encoder as a dynamical system, developed in order to understand better the relationship between the interleaver in a PCCC and the constituent encoders. The results in this chapter are based on the analysis of a PCCC whose constituent encoders are represented by a state variable model of a discrete dynamical system.

3.1.1 Recursive Convolutional Encoder as a Dynamical System

For the encoder shown in Fig. 7, the state variable model consists of a state vector $\underline{v}$, containing the delay states v_1, v_2, v_3, and the input m that is the next input sequence

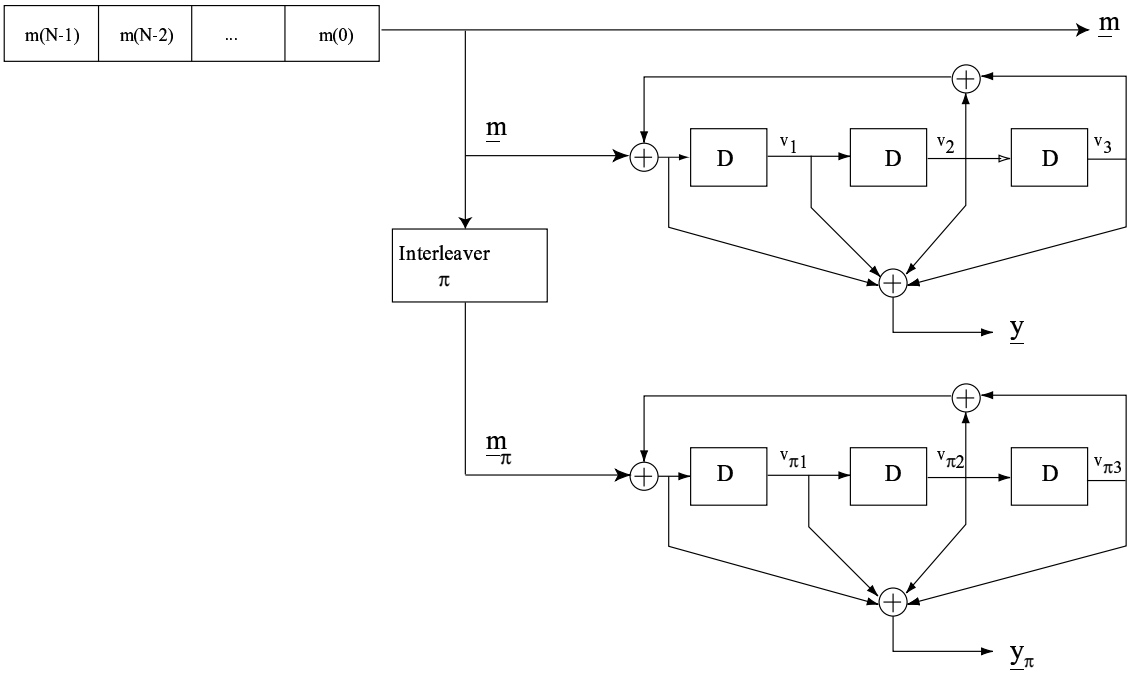

Figure 7: Parallel Concatenated Convolutional Coding Scheme

bit to be encoded. The output, y, could also be included in the state variable model, but it is not necessary for this discussion. Note that since this is a binary coding scheme all addition is mod 2.

$$\begin{bmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{bmatrix} \oplus m(n) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{4}$$

In practice, there are a finite number, N, of input sequence bits to be encoded. We can think of these bits as being placed into a length-N vector $\underline{m}$ to be queued into the encoder as shown in Fig. 7. As each bit is shifted to the right into the encoder, we place a 0 into the left-most element of $\underline{m}$ until all N input sequence bits have been encoded and $\underline{m}$ is a vector of zeros. This leads to an interpretation of the encoder as an autonomous dynamical system where the delay states $\underline{v}$ and input sequence vector

$\underline{m}$ are the state variables:

$$\begin{bmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \\ m_1(n+1) \\ \vdots \\ m_{N-2}(n+1) \\ m_{N-1}(n+1) \\ m_N(n+1) \end{bmatrix} = \begin{bmatrix} 0\ 1\ 1\ 1\ 0 \cdots 0\ 0 \\ 1\ 0\ 0\ 0\ 0 \cdots 0\ 0 \\ 0\ 1\ 0\ 0\ 0 \cdots 0\ 0 \\ 0\ 0\ 0\ 0\ 1 \cdots 0\ 0 \\ \vdots \qquad\qquad \ddots \\ 0\ 0\ 0\ 0\ 0 \cdots 1\ 0 \\ 0\ 0\ 0\ 0\ 0 \cdots 0\ 1 \\ 0\ 0\ 0\ 0\ 0 \cdots 0\ 0 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \\ m_1(n) \\ \vdots \\ m_{N-2}(n) \\ m_{N-1}(n) \\ m_N(n) \end{bmatrix} = A \begin{bmatrix} \underline{v}(n) \\ \underline{m}(n) \end{bmatrix}$$

Furthermore, we notice that after N time steps the state of the system is given by

$$\begin{bmatrix} \underline{v}(N) \\ \underline{m}(N) \end{bmatrix} = A \cdot \begin{bmatrix} \underline{v}(N-1) \\ \underline{m}(N-1) \end{bmatrix} = A^N \cdot \begin{bmatrix} \underline{v}(0) \\ \underline{m}(0) \end{bmatrix} \tag{5}$$

where

$$\underline{v}(0) = \begin{bmatrix} 0\ 0\ 0 \end{bmatrix}' \tag{6}$$

$$\underline{m}(N) = \begin{bmatrix} 0\ 0 \cdots 0 \end{bmatrix}' \tag{7}$$

and

$$A^N = \begin{bmatrix} x\ x\ x\ 1\ 0\ 0\ 1\ 1\ 1\ 0 \cdots 1\ 0\ 0\ 1\ 1\ 1\ 0 \\ x\ x\ x\ 0\ 0\ 1\ 1\ 1\ 0\ 1 \cdots 0\ 0\ 1\ 1\ 1\ 0\ 1 \\ x\ x\ x\ 0\ 1\ 1\ 1\ 0\ 1\ 0 \cdots 0\ 1\ 1\ 1\ 0\ 1\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \cdots 0\ 0\ 0\ 0\ 0\ 0\ 0 \\ \vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots \qquad \vdots \qquad\qquad \vdots \\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \cdots 0\ 0\ 0\ 0\ 0\ 0\ 0 \end{bmatrix}. \tag{8}$$

The first three rows of the matrix A^N from column 4 to column $N+3$ consist of a pattern of bits that repeats periodically. The remainder of the rows of A^N are filled entirely with zeros. This is true of A^M for all $M \geq N$. Careful observation shows that these bit patterns, which have the same period as the feedback polynomial of the encoder, correspond to what we refer to as the "impulse responses of the delay states of the encoder," as seen in Fig. 8. The rows of A^M containing the delay state impulse responses will change as A^M is iterated; however, since the rows of A^M are linear combinations of the rows of A^N, the delay state impulse responses of A^M will always be elements of the vector space spanned by the rows of the delay state impulse responses of A^N.

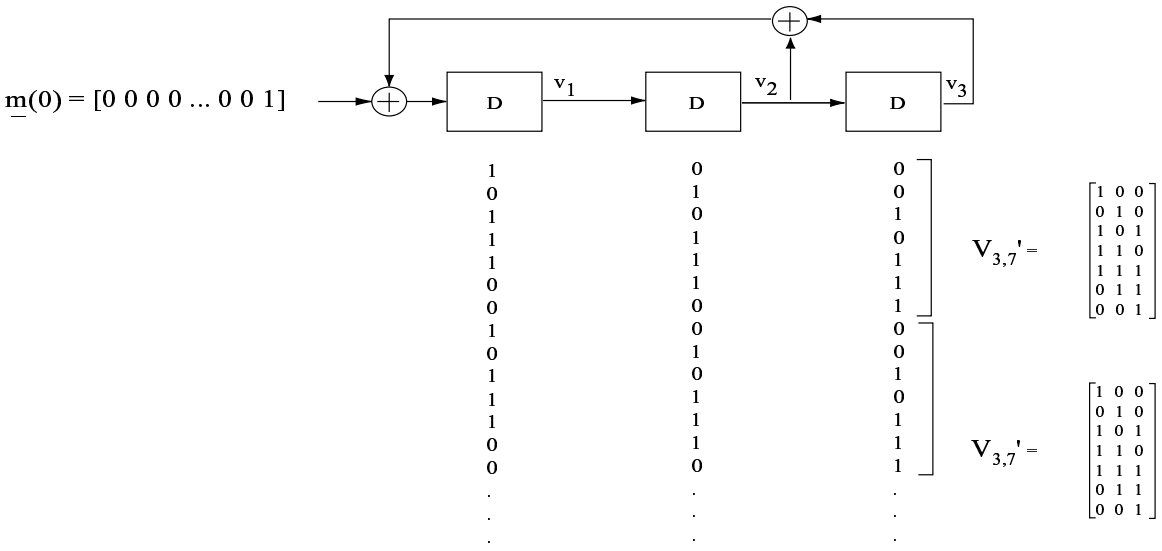

Figure 8: Impulse response of the delay states of the encoder

In Fig. 8 we introduce notation for the matrix $V_{k,ln}$ that contains the k delay state impulse responses over n length-l periods of the encoder. The encoder in Fig. 8 has $k = 3$ delay states and an impulse response with period $l = 7$. Thus, $V_{3,7}$ is a matrix containing the three delay state impulse responses over a single length-7 period of the impulse response of the encoder.

An important relationship exists between the impulse responses of the delay states and a class of error patterns called terminated errors. This relationship leads to new insights into interleaver performance and design that we will discuss in Section 3.1.2.

3.1.2 Fundamental Properties of Interleavers

A terminating input sequence is an input sequence that begins with the encoder in the all-zero state (i.e., with $\underline{v}(0) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}'$) and returns the encoder to the all-zero state after its last non-zero bit has been encoded. PCCCs employ recursive convolutional encoders, which have infinite impulse responses, as their constituent encoders. Therefore, most input sequences to the encoders generate parity sequences that are not self-terminated (i.e., the parity sequence is terminated artificially at

the end of the code block, but would not terminate given an infinite length code block) in one or both of the encoders. As a result, most codewords in a PCCC have high Hamming weight. The error patterns most likely to be uncorrected by a maximum likelihood decoding algorithm are those corresponding to low Hamming weight codewords in both of the constituent encoders. These codewords are generated by the terminating input sequences that remain as terminating input sequences after interleaving.

One of the interesting properties of a terminating input sequence is that a row vector containing a length-N terminating input sequence is always (mod 2) orthogonal to each of the delay state impulse responses of that encoder. This property allows us to make some fundamental observations about recursive convolutional encoders and the terminating input sequences associated with them.

We will use the notation $\underline{d}_1$,$\underline{d}_2$, and $\underline{d}_3$ for the length-N delay state impulse responses of the encoder in Fig. 8 such that[1]

$$\begin{aligned}\begin{bmatrix}\underline{d}_1\\ \underline{d}_2\\ \underline{d}_3\end{bmatrix} &= \begin{bmatrix} V_{3,7} \cdots V_{3,7}\end{bmatrix}\\ &= \begin{bmatrix}1\,0\,0\,1\,1\,1\,0\cdots 1\,0\,0\,1\,1\,1\,0\\ 0\,0\,1\,1\,1\,0\,1\cdots 0\,0\,1\,1\,1\,0\,1\\ 0\,1\,1\,1\,0\,1\,0\cdots 0\,1\,1\,1\,0\,1\,0\end{bmatrix}.\end{aligned} \tag{9}$$

A row vector containing a terminating input sequence $\underline{m}_t$, given by

$$\underline{m}_t = \begin{bmatrix} m_t(0)\ m_t(1)\cdots m_t(N-1)\end{bmatrix},$$

and a delay state impulse response

$$\underline{d}_n = \begin{bmatrix} d_n(0)\ d_n(1)\cdots d_n(N-1)\end{bmatrix}, 0 \le n \le k,$$

are orthogonal if they satisfy:

$$\underline{d}_n \cdot \underline{m}'_t = d_n(0)\, m_t(0) \oplus d_n(1)\, m_t(1) \oplus \cdots \oplus d_n(N-1)\, m_t(N-1) = 0$$

[1]For ease of notation, we assume here that N is an integer multiple of the period of the feedback polynomial of the encoder, but this is not a necessary condition for our argument.

where $\oplus$ denotes mod 2 addition. By definition, a terminating input sequence will leave the encoder in the zero-state after all its bits have been encoded. After a length-N terminating input sequence has been encoded, the state of the encoder is given by $\underline{v}(N) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}'$. Simplifying (5) through (8) and incorporating notation from (9) for the delay state impulse responses, we get the following expression for the state of the encoder after a length-N terminating input sequence has been encoded:

$$\begin{aligned} \begin{bmatrix} v_1(N) \\ v_2(N) \\ v_3(N) \end{bmatrix} &= \begin{bmatrix} 1\,0\,0\,1\,1\,1\,0 \cdots 1\,0\,0\,1\,1\,1\,0 \\ 0\,0\,1\,1\,1\,0\,1 \cdots 0\,0\,1\,1\,1\,0\,1 \\ 0\,1\,1\,1\,0\,1\,0 \cdots 0\,1\,1\,1\,0\,1\,0 \end{bmatrix} \begin{bmatrix} m_t(0) \\ \vdots \\ m_t(N-1) \end{bmatrix} \\ &= \begin{bmatrix} \underline{d}_1 \\ \underline{d}_2 \\ \underline{d}_3 \end{bmatrix} \begin{bmatrix} m_t(0) \\ \vdots \\ m_t(N-1) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \end{aligned} \tag{10}$$

which shows that $\underline{m}_t'$ is (mod 2) orthogonal to the impulse response of each delay state of the encoder. This observation leads to two fundamental properties of recursive convolutional encoders:

Lemma 1 *For a recursive convolutional encoder with k delay states, out of all 2^N possible input sequence patterns where N is the length of the input sequence vector $\underline{m}$, at least 2^{N-k} of these are terminating input sequences.*

Proof. The set M of length-N input sequences forms an N-dimensional vector space over $GF(2)$. Since each coordinate of an input sequence can be either a 1 or a 0, the cardinality of an N-dimensional vector space M is 2^N.

A terminating input sequence $\underline{m}_t$ is, as shown in (10), an input sequence whose transpose is orthogonal to each of the delay state impulse responses of the encoder:

$$\begin{bmatrix} \underline{d}_1 \\ \underline{d}_2 \\ \vdots \\ \underline{d}_k \end{bmatrix} \underline{m}_t = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Thus, M_t, the set of all length-N terminating input sequences $\underline{m}_t$, is the dual space to a vector subspace of M that is spanned by the transpose of the delay state impulse

response vectors $\left\{ \underline{d}'_1, \underline{d}'_2, \cdots \underline{d}'_k \right\}$. If the k delay state impulse response vectors are linearly independent, then the dimension of M_t is $N-k$. If the delay state impulse response vectors are not linearly independent, then the dimension of M_t is greater than $N-k$. As a result, there are at least 2^{N-k} terminating input sequences for an encoder with k delay states. ■

Lemma 2 *For a recursive convolutional encoder with k linearly independent delay state impulse responses, out of all 2^{N-k} possible terminating input sequences, it is possible to interleave at most $2^{N-k}\left(\frac{2^k-1}{2^k}\right)$ terminating input sequences into non-terminating input sequences.*

Proof. After interleaving, a terminating input sequence $\underline{m}_t$ is given by the vector $P \cdot \underline{e}_t$ where P is a permutation matrix. The terminating input sequences $\widetilde{\underline{m}}_t$ that are not interleaved into non-terminating input sequences satisfy:

$$\begin{bmatrix} \underline{d}_1 \\ \underline{d}_2 \\ \vdots \\ \underline{d}_k \end{bmatrix} P \cdot \widetilde{\underline{m}}_t = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Thus, the non-interleaved input sequences $\widetilde{\underline{m}}_t$ constitute the dual space of the vector subspace spanned by the vectors $\left\{ \underline{d}'_1, \underline{d}'_2, \cdots \underline{d}'_k, P' \cdot \underline{d}'_1, P' \cdot \underline{d}'_2, \cdots P' \cdot \underline{d}'_k \right\}$. If the k delay state impulse responses and the k permuted delay state impulse responses are linearly independent, then the dual space to the vector space spanned by these $2k$ basis vectors has dimension $N-k-k$. This $N-k-k$ dimension vector space consists of 2^{N-k-k} vectors. Thus, there are at least 2^{N-k-k} terminating input sequences that cannot be interleaved into non-terminating input sequences. ■

Note that significantly different proofs of Lemma 1 and Lemma 2 were given by Khandani in [30] and [31], and that the results presented here were developed independently and published in [32]. Furthermore, we will show in Section 3.2 that the proofs presented here lead to a low-complexity method for designing interleavers

that satisfy the upper bound set forth in Lemma 2 on the number of terminating input sequences that can be interleaved into non-terminating input sequences. We will also show how to choose these interleavers to achieve significant gains in the BER performance of PCCCs.

3.2 Sub-vector Interleaving

By rewriting (10) in terms of the matrix $V_{3,7}$ defined in Section 3.1.1 and shown in Fig. 8, we see that for the encoder with three delay states shown in Fig. 7 coupled with a length-N interleaver where N is a multiple of $m = 7$, the period of the feedback polynomial of the encoder, all terminating input sequences $\underline{m}_t$ must satisfy

$$\begin{bmatrix} \underbrace{\begin{matrix} 1&0&0&1&1&1&0 \\ 0&0&1&1&1&0&1 \\ 0&1&1&1&0&1&0 \end{matrix}}_{V_{3,7}} & \cdots & \underbrace{\begin{matrix} 1&0&0&1&1&1&0 \\ 0&0&1&1&1&0&1 \\ 0&1&1&1&0&1&0 \end{matrix}}_{V_{3,7}} \end{bmatrix} \cdot \underline{m}_t = \begin{bmatrix} V_{3,7} & \cdots & V_{3,7} \end{bmatrix} \cdot \underline{m}_t$$

$$= \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

For the recursive convolutional encoder being considered here, there exist 1344 length-7 interleavers that can be used to satisfy the upper bound given in Lemma 2, and these are easily found by exhaustive computer search over the 7! possible length-7 interleavers. We will refer to such an interleaver as $P_{opt,7}$, since it is optimal in the sense that it can be used to interleave the maximum fraction of length-7 terminating input sequences into non-terminating input sequences in a PCCC.

Expressing the terminating input sequence $\underline{m}_t$ as the vertical concatenation of length-7 sub-vectors, we see in (11) that the (mod 2) summation of these sub-vectors

is also a terminating input sequence:

$$\begin{aligned}\begin{bmatrix} V_{3,7} & \cdots & V_{3,7} \end{bmatrix} \cdot \underline{m}_t &= \begin{bmatrix} V_{3,7} & \cdots & V_{3,7} \end{bmatrix} \cdot \begin{bmatrix} \underline{m}_{t_1} \\ \vdots \\ \underline{m}_{t_{\frac{N}{7}}} \end{bmatrix} \\ &= [V_{3,7}] \begin{bmatrix} \underline{m}_{t_1} \oplus \underline{m}_{t_2} \oplus \cdots \oplus \underline{m}_{t_{\frac{N}{7}}} \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \end{aligned} \tag{11}$$

The dimensions of $P_{opt,7}$ correspond to the period of the feedback polynomial of our recursive convolutional encoder. As a result, we can place our length-7 interleaver on the diagonal of an $N \times N$ (choosing N to be a multiple of 7) permutation matrix to create a length-N interleaver $P_{opt,N}$ that shares the properties of $P_{opt,7}$ in that it interleaves the maximum fraction of length-N terminating input sequences into non-terminating input sequences:

$$\begin{aligned}&\begin{bmatrix} V_{3,7} & \cdots & V_{3,7} \end{bmatrix} \cdot \begin{bmatrix} P_{opt,7} & 0 & \cdots & 0 \\ 0 & P_{opt,7} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & P_{opt,7} \end{bmatrix} \cdot \begin{bmatrix} \underline{m}_{t_1} \\ \vdots \\ \underline{m}_{t_{\frac{N}{7}}} \end{bmatrix} \\ &= [V_{3,7}] \cdot P_{opt,7} \cdot \begin{bmatrix} \underline{m}_{t_1} \oplus \underline{m}_{t_2} \oplus \cdots \oplus \underline{m}_{t_{\frac{N}{7}}} \end{bmatrix}. \end{aligned} \tag{12}$$

We refer to this method of interleaving as "sub-vector interleaving." Sub-vector interleavers effectively permute the columns of a $d \times \frac{N}{d}$ matrix, where d is the length of the sub-vectors being considered and N is the length of the interleaver, with the input sequence read in row-wise.

We now need to consider interleaving not just the maximum fraction of terminating input sequences, but those input sequences that generate the codewords that are most significant in terms of their effect on the BER performance of the PCCC. We will discuss these error patterns and their contribution to the BER in Section 3.4.

3.3 Dual Trellis Termination with Minimum Length Tail Sequence for PCCCs

In this section we will show that it is possible to append a single terminating tail sequence to the input of the PCCC that causes each of the constituent encoders to be terminated in the zero state and allows for sub-vector interleaving to occur without affecting the encoder termination. Note that other researchers have reported similar results for encoder termination using other types of interleavers in [29], [5], and [13].

3.3.1 Dynamical System Representation of a PCCC with a sub-vector Interleaver

To compute a tail sequence that will terminate the constituent encoders of a PCCC, it is helpful to represent the entire PCCC as a dynamical system. This dynamical system takes as its input an entire length-η sub-vector of the input sequence m at each iteration.

Thus, the dynamical system representation of the top encoder alone in the PCCC shown in Fig. 7 employing a length-14 sub-vector interleaver is given by:[2]

$$\begin{bmatrix} \\ v_1(n+14) \\ v_2(n+14) \\ \underbrace{v_3(n+14)}_{\underline{v}(n+14)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{bmatrix} + \begin{bmatrix} V_{3,7} & V_{3,7} \end{bmatrix} \begin{bmatrix} m(n) \\ \vdots \\ m(n+14) \end{bmatrix}. \tag{13}$$

The dynamical system representation of the bottom encoder alone in the PCCC shown in Fig. 7 employing a length-14 sub-vector interleaver, P, is given by:

$$\begin{bmatrix} \\ v_{\pi 1}(n+14) \\ v_{\pi 2}(n+14) \\ \underbrace{v_{\pi 3}(n+14)}_{\underline{v}_\pi(n+14)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{\pi 1}(n) \\ v_{\pi 2}(n) \\ v_{\pi 3}(n) \end{bmatrix} + \begin{bmatrix} \underbrace{V_{3,7} & V_{3,7}}_{V_{3,14}} \end{bmatrix} \cdot P \cdot \begin{bmatrix} m(n) \\ \vdots \\ m(n+14) \end{bmatrix}, \tag{14}$$

[2]The choice of sub-vector length will be explained in Sec. 3.4

where

$$V_{3,14} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

and

$$P = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{15}$$

(this chioce of sub-vector interleaver, P, will be explained in Section 3.4), which gives us

$$\begin{aligned} [V_{3,14}] \cdot P &= \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \\ &= \left[V_{\pi_{3,14}} \right] \end{aligned}$$

If we combine (13) and (14) and incorporate the input sequence, $\underline{m}$, into the state equations for the dynamical system, the result is an autonomous dynamical system representation of the entire PCCC, including the sub-vector interleaver:

$$\begin{bmatrix} \underline{v}\,(n+14) \\ \underline{v}_{\pi}\,(n+14) \\ \underline{m}\,(n+14) \end{bmatrix} = \begin{bmatrix} I & 0 & V_{3,14} & 0 \\ 0 & I & V_{\pi_{3,14}} & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \underline{v}\,(n) \\ \underline{v}_{\pi}\,(n) \\ \underline{m}\,(n) \end{bmatrix},$$

where $\underline{v}\,(n)$ contains the delay states of the top encoder at time n, and $\underline{v}_{\pi}\,(n)$ contains the delay states of the lower encoder at time n. The vector $\underline{m}(0)$ contains the entire length-N input sequence to the PCCC. As the autonomous dynamical system iterates, length-14 sub-vectors of $\underline{m}$ are encoded and then shifted out the top of $\underline{m}$, while a

length-14 zero vector shifts into the bottom of $\underline{m}$. Therefore, $\underline{m}(N)$ is a length-N zero vector.

If the autonomous dynamical system is iterated $\frac{N}{14}$ times, the final state of the constituent encoders at time $n = N$ can be expressed as a mapping of the input sequence $\underline{m}$ at $n = 0$:

$$\begin{bmatrix} \underline{v}\,(N) \\ \underline{v}_{\pi}\,(N) \end{bmatrix} = \begin{bmatrix} V_{3,14} & V_{3,14} & \cdots & V_{3,14} & V_{3,14} \\ V_{\pi_{3,14}} & V_{\pi_{3,14}} & \cdots & V_{\pi_{3,14}} & V_{\pi_{3,14}} \end{bmatrix} \underline{m}\,(0)\,,$$

which implies that

$$\begin{bmatrix} \underline{v}\,(N) \\ \underline{v}_{\pi}\,(N) \end{bmatrix} = \begin{bmatrix} V_{3,14} \\ V_{\pi_{3,14}} \end{bmatrix} \underline{m}_{\Sigma},$$

where $\underline{m}_{\Sigma}\,(0)' = \begin{bmatrix} m_{\Sigma_1} & m_{\Sigma_2} & \cdots & m_{\Sigma_{13}} & m_{\Sigma_{14}} \end{bmatrix}$ is the mod 2 summation of length-14 sub-vectors of the input sequence contained in $\underline{m}\,(0)$.

If $\underline{v}\,(N) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$ and $\underline{v}_{\pi}\,(N) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$, then the input sequence to the PCCC is a terminating input sequence, $\underline{m}_t$. The rows of the matrices $V_{3,14}$ and $V_{\pi_{3,14}}$ and $\underline{m}_{\Sigma_t}$, the mod 2 summation of the length-14 sub-vectors of $\underline{m}_t$, are mod 2 orthogonal:

$$\begin{bmatrix} V_{3,14} \\ V_{\pi_{3,14}} \end{bmatrix} \underline{m}_{\Sigma_t} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This yields to the following relationship between the elements of the vector $\underline{m}_{\Sigma_t}$:

$$\begin{bmatrix} m_{\Sigma_{t1}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t8}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t12}} \oplus m_{\Sigma_{t13}} \\ m_{\Sigma_{t3}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t7}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t12}} \oplus m_{\Sigma_{t14}} \\ m_{\Sigma_{t2}} \oplus m_{\Sigma_{t3}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t9}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t13}} \\ m_{\Sigma_{t1}} \oplus m_{\Sigma_{t3}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t8}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t12}} \oplus m_{\Sigma_{t14}} \\ m_{\Sigma_{t1}} \oplus m_{\Sigma_{t3}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t7}} \oplus m_{\Sigma_{t8}} \oplus m_{\Sigma_{t9}} \oplus m_{\Sigma_{t10}} \\ m_{\Sigma_{t2}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t7}} \oplus m_{\Sigma_{t8}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t12}} \oplus m_{\Sigma_{t13}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \qquad (16)$$

A length-6 terminating tail sequence,

$$\underline{m}_{tail} = \begin{bmatrix} m\,(N-5) & m\,(N-4) & \cdots & m\,(N-1) & m\,(N) \end{bmatrix},$$

appended to a length-$(N-6)$ input sequence leads to the following relationship between $\underline{m}_{\Sigma}$ and $\underline{m}_{\Sigma_t}$:

$$\begin{bmatrix} m_{\Sigma_1} \\ m_{\Sigma_2} \\ \vdots \\ m_{\Sigma_8} \\ m_{\Sigma_9} \oplus m(N-5) \\ m_{\Sigma_{10}} \oplus m(N-4) \\ m_{\Sigma_{11}} \oplus m(N-3) \\ m_{\Sigma_{12}} \oplus m(N-2) \\ m_{\Sigma_{13}} \oplus m(N-1) \\ m_{\Sigma_{14}} \oplus m(N) \end{bmatrix} = \begin{bmatrix} m_{\Sigma_{t1}} \\ m_{\Sigma_{t2}} \\ \vdots \\ m_{\Sigma_{t8}} \\ m_{\Sigma_{t9}} \\ m_{\Sigma_{t10}} \\ m_{\Sigma_{t11}} \\ m_{\Sigma_{t12}} \\ m_{\Sigma_{t13}} \\ m_{\Sigma_{t14}} \end{bmatrix}. \tag{17}$$

Substituting (16) into (17) yields six linearly independent equations which can be solved for $\underline{m}_{tail}$:

$$\begin{bmatrix} 0&0&1&1&1&0 \\ 0&1&1&1&0&1 \\ 1&1&1&0&1&0 \\ 0&0&1&1&0&1 \\ 1&1&0&0&0&0 \\ 0&1&0&1&1&0 \end{bmatrix} \begin{bmatrix} m(N-5) \\ m(N-4) \\ m(N-3) \\ m(N-2) \\ m(N-1) \\ m(N) \end{bmatrix} =$$

$$\begin{bmatrix} 1&0&0&1&1&1&0&1&0&0&1&1&1&0 \\ 0&0&1&1&1&0&1&0&0&1&1&1&0&1 \\ 0&1&1&1&0&1&0&0&1&1&1&0&1&0 \\ 1&0&1&0&1&1&0&1&0&0&1&1&0&1 \\ 1&0&1&1&1&0&1&1&1&1&0&0&0&0 \\ 0&1&0&0&1&1&1&1&0&1&0&1&1&0 \end{bmatrix} \begin{bmatrix} m_{\Sigma_1} \\ m_{\Sigma_2} \\ \vdots \\ m_{\Sigma_{12}} \\ m_{\Sigma_{13}} \\ m_{\Sigma_{14}} \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} m(N-5) \\ m(N-4) \\ m(N-3) \\ m(N-2) \\ m(N-1) \\ m(N) \end{bmatrix} = \begin{bmatrix} 0&1&1&0&0&0&1&0&0&0&0&1&1&1 \\ 1&0&1&0&1&1&0&0&1&1&1&1&0&0 \\ 0&1&1&0&1&0&0&0&0&1&1&0&0&1 \\ 1&0&0&1&0&1&1&0&1&0&0&0&0&1 \\ 1&0&1&0&0&1&1&1&0&1&0&0&1&1 \\ 0&0&1&0&1&1&0&0&1&0&0&0&0&0 \end{bmatrix} \begin{bmatrix} m_{\Sigma_1} \\ m_{\Sigma_2} \\ \vdots \\ m_{\Sigma_{12}} \\ m_{\Sigma_{13}} \\ m_{\Sigma_{14}} \end{bmatrix}.$$

Sub-vector interleaving effectively permutes the columns of a matrix with the information sequence, m, read in row-wise. Permutations along each of the columns of this matrix do not affect the termination of the encoder, since the vector $\underline{m}_{\Sigma}$ which contains the mod 2 sum of the elements in each of the columns is unaffected.

3.4 Lowering the Error Floor of Turbo Codes

We saw in Section 2.4.1.1 that at low SNRs, weight-2 terminating input sequences corresponding to codewords that generate parity sequences with the lowest possible Hamming weight determine the BER of the PCCC. In the particular case of a PCCC employing encoders as in Fig. 7, this codeword is generated by the input sequences

$$\underline{m} = \underline{m}_{\pi} = \left[1\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \right]$$

that generates the parity sequence outputs

$$\underline{y} = \underline{y}_{\pi} = \left[1\ 1\ 0\ 1\ 1\ 1\ 0\ 1 \right]$$

The probability of such an error pattern corresponding to this codeword being uncorrected in the decoder is given by (1) evaluated with $d = 2$ and $p = d + p_1 + p_2 = 14$, l_1 and $l_2 = 8$, and $R_c = \frac{1}{2}$:

$$P_b\left(e \mid \underline{e}_{2,14}\right) \approx \left(\tfrac{2 \cdot 2!}{N^3}\right) (N-7)^2 \, Q\left(\sqrt{14 \cdot \tfrac{\varepsilon_b}{N_0}}\right) \tag{18}$$

At asymptotically high SNRs, error patterns corresponding to the free distance codeword are the most likely to be uncorrected in the decoder. The free distance codeword for this encoder is generated by the input sequence

$$\underline{m} = \underline{m}_{\pi} = \left[1\ 1\ 1\ 0\ 1 \right]$$

that generates the parity sequence outputs

$$\underline{y} = \underline{y}_{\pi} = \left[1\ 0\ 0\ 0\ 1 \right]$$

The probability of an error resulting from an error pattern corresponding the free distance codeword is given by (1) evaluated with $d = 4$ and $p = d + p_1 + p_2 = 8$, l_1 and $l_2 = 5$, and $R_c = \frac{1}{2}$:

$$P_b\left(e \mid \underline{e}_{4,8}\right) \approx \left(\tfrac{4 \cdot 4!}{N^5}\right) (N-4)^2 \, Q\left(\sqrt{8 \cdot \tfrac{\varepsilon_b}{N_0}}\right) \tag{19}$$

The sum of the error probabilities in (18) and (19) forms a lower bound on the BER for our PCCC (see Fig. 11).

For our interleaver design to improve the performance of the PCCC over a wide range of SNRs, we must target the weight-2 terminating input sequences that generate low-weight parity sequences in addition to targeting the error patterns corresponding to the free distance codeword. To take into account the weight-2 terminating input sequences that correspond to low-weight codewords, we increased the sub-vector length to 14 and searched over length-14 interleavers for a suitable sub-vector interleaver. We were able to find many length-14 interleavers that permuted all weight-2 terminating input sequences with span < 14 into non-terminating input sequences. Within this collection of interleavers, we found several that interleaved all length-14 cyclic shifts of the error patterns corresponding to the free distance codewords into higher weight error patterns.

We used the permutation P in (15) as the length-14 sub-vector interleaver to create a length-14000 interleaver as in (12) that also satisfied the design criteria described in the previous paragraph. This sub-vector interleaver effectively permutes the columns of a 1000×14 matrix with information bits read in row-wise. Since some terminating input sequences will (by Lemma 2) be interleaved into similar low-weight terminating input sequences after the columns are permuted, we can reduce the probability that they are not corrected in the decoder by interleaving randomly over each column of the matrix. This increases the expected weight of the parity sequences of the terminating input sequences that are not interleaved into non-terminating input sequences by our sub-vector interleaver and allows us to scale up the interleaver to any length desired while achieving the BER reduction expected with increased interleaver length (i.e., the interleaver gain).

3.4.1 Distance Spectrum Analysis

Using the method outlined in [22], we are able to compute the weight distribution of codewords for a PCCC employing a particular length-N interleaver. Performing this analysis over a large number (≈ 10000) of interleavers gives us an average weight distribution for a PCCC. By doing this we have found that sub-vector interleavers increase the average free distance codeword weight of a PCCC. Fig. 9 compares the average free distance codeword weight of a PCCC employing either a random or a sub-vector interleaver as a function of interleaver length.[3]

Our weight distribution analysis allowed us to compute the BER contribution of input sequences as a function of their Hamming weight. In Fig. 10 we plot the BER contribution of the most significant (in terms of BER contribution) codewords generated by weight-2 terminating input sequences assuming a length-14000 average random interleaver. For reference, we also plot the BER contribution of the free distance codeword (generated by a weight-4 terminating input sequence in this case). The BER contribution most significant codewords can be eliminated by sub-vector interleaving, and the free distance codeword weight can be increased. Results of a simulation of a PCCC employing the two types of interleaving will be shown in Sec. 3.4.2.

3.4.2 Simulation Results

The simulation results of the PCCC employing a length-14000 sub-vector interleaver are reported in Fig. 11 along with the results of the simulation of the same scheme

[3]Note that the interleaver lengths are not quite equal for the sub-vector interleaved and randomly interleaved PCCC. Since the sub-vector interleaved PCCC can be terminated with a single tail sequence while the randomly interleaved PCCC requires a separate tail sequence to terminate each of the constituent encoders, we chose to keep the total length of the input sequence plus the two parity sequences, including tail bits, equal rather than the interleaver lengths. This allowed for a more accurate comparison between the average free distance codeword weights for the two types of interleavers. Thus, the interleaver length for the sub-vector interleaved PCCC is slightly longer than that of the randomly interleaved PCCC.

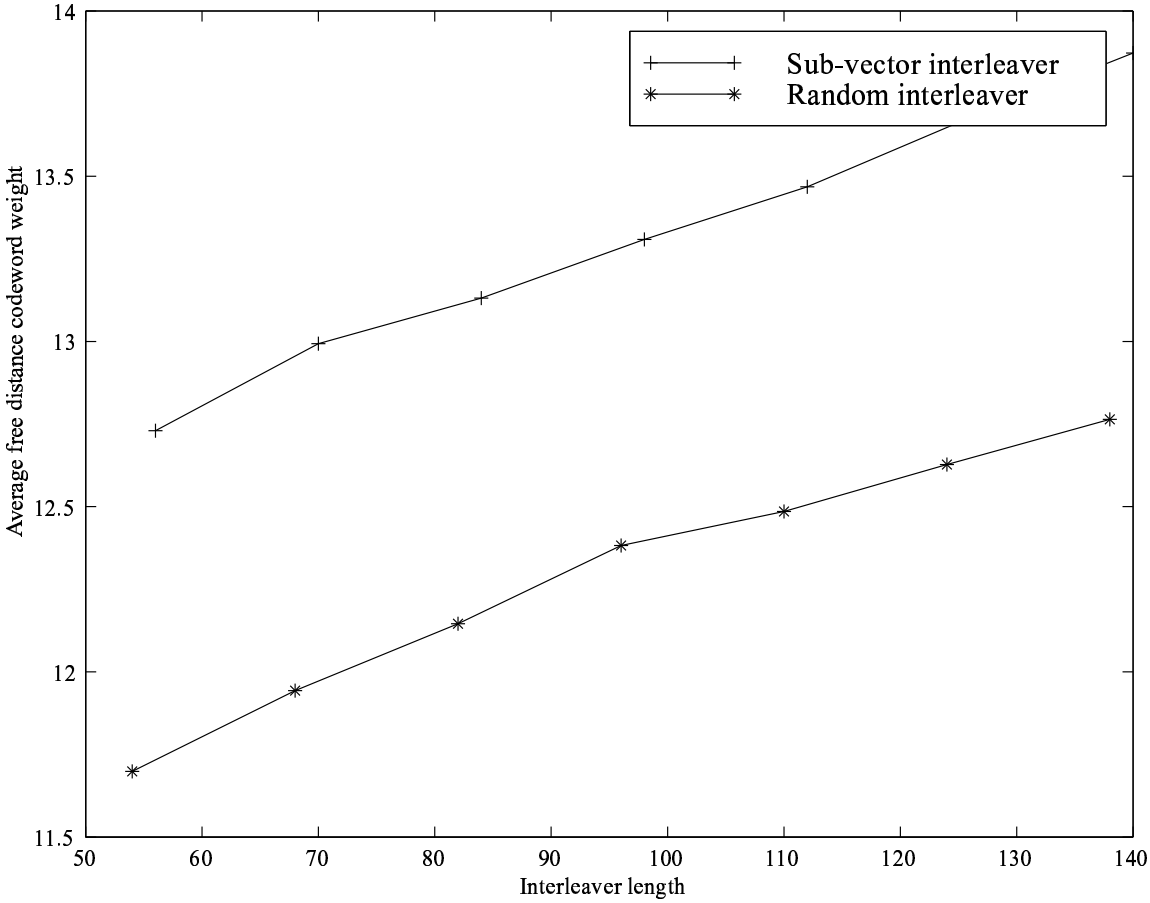

Figure 9: Average free distance codeword weight as a function of interleaver length for a PCCC employing encoders shown in Fig. 7 and a random interleaver and the same scheme employing an sub-vector interleaver.

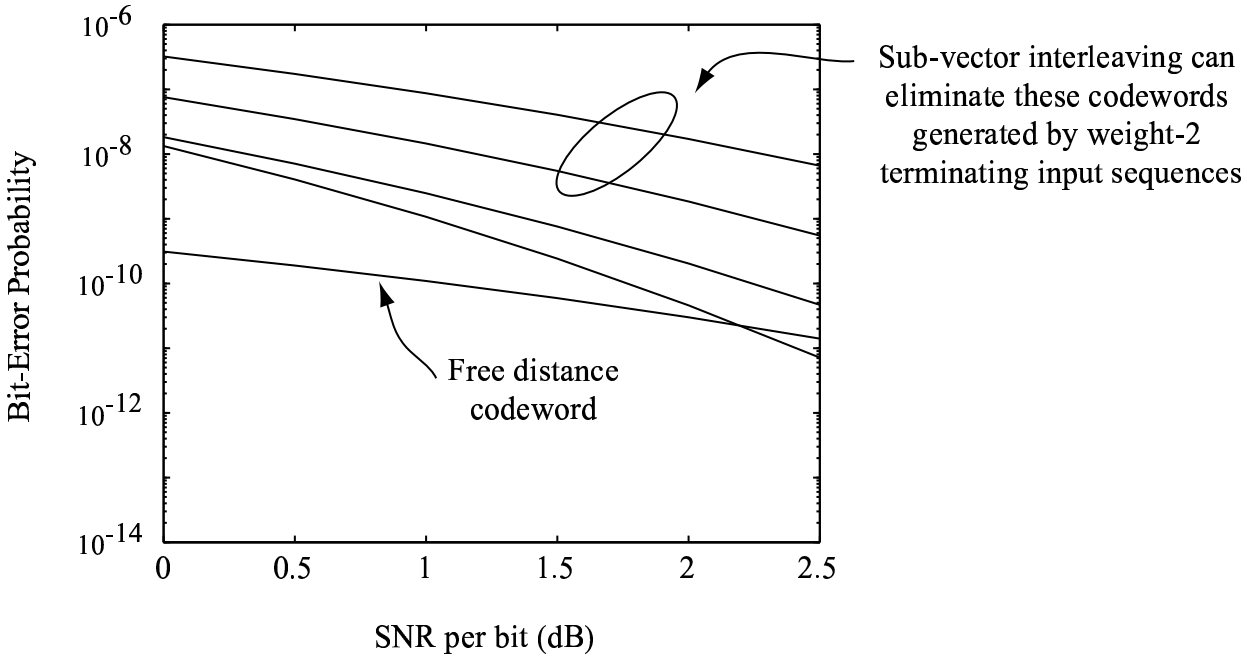

Figure 10: BER contribution of selected weight-2 terminating input sequences and the weight-3 terminating input sequence that generates the free distance codeword for the PCCC in Fig. 7 and a length-14000 random interleaver. Sub-vector interleaving eliminates the BER contribution of the two most significant weight-2 terminating input sequences.

employing a length-14000 random interleaver. We calculated that the lowest Hamming weight of a codeword generated by a weight-2 terminating input sequence for the PCCC employing a length-14000 sub-vector interleaver is the codeword generated by the input sequence, $\underline{m}$, and interleaved into the identical input sequence, $\underline{m}_\pi$:

$$\underline{m} = \underline{m}_\pi = \left[1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\right].$$

Both $\underline{m}$ and $\underline{m}_\pi$ generate identical parity sequences

$$\underline{y} = \underline{y}_\pi = \left[1\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1\right].$$

Thus, the lowest Hamming weight of a codeword generated by a weight-2 terminating input sequence is increased to $p = 22$ for sub-vector interleaving from $p = 14$ for random interleaving.

The weight of the free distance codeword for sub-vector interleaving, which is generated by the input sequence

$$\underline{m} = \left[0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\right]$$

that is interleaved into

$$\underline{m}_\pi = \left[0\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\right]$$

and generates the parity sequences

$$\underline{y} = \left[0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\right]$$

and

$$\underline{y}_\pi = \left[0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\right],$$

respectively, is increased to $p = 10$ from $p = 8$ for the average random interleaver. The lower bounds for both random interleaving and sub-vector interleaving are plotted alongside the simulated performance curves. These curves show that the probability of a bit error converges to these asymptotic lower bounds for a length-14000 interleaver.

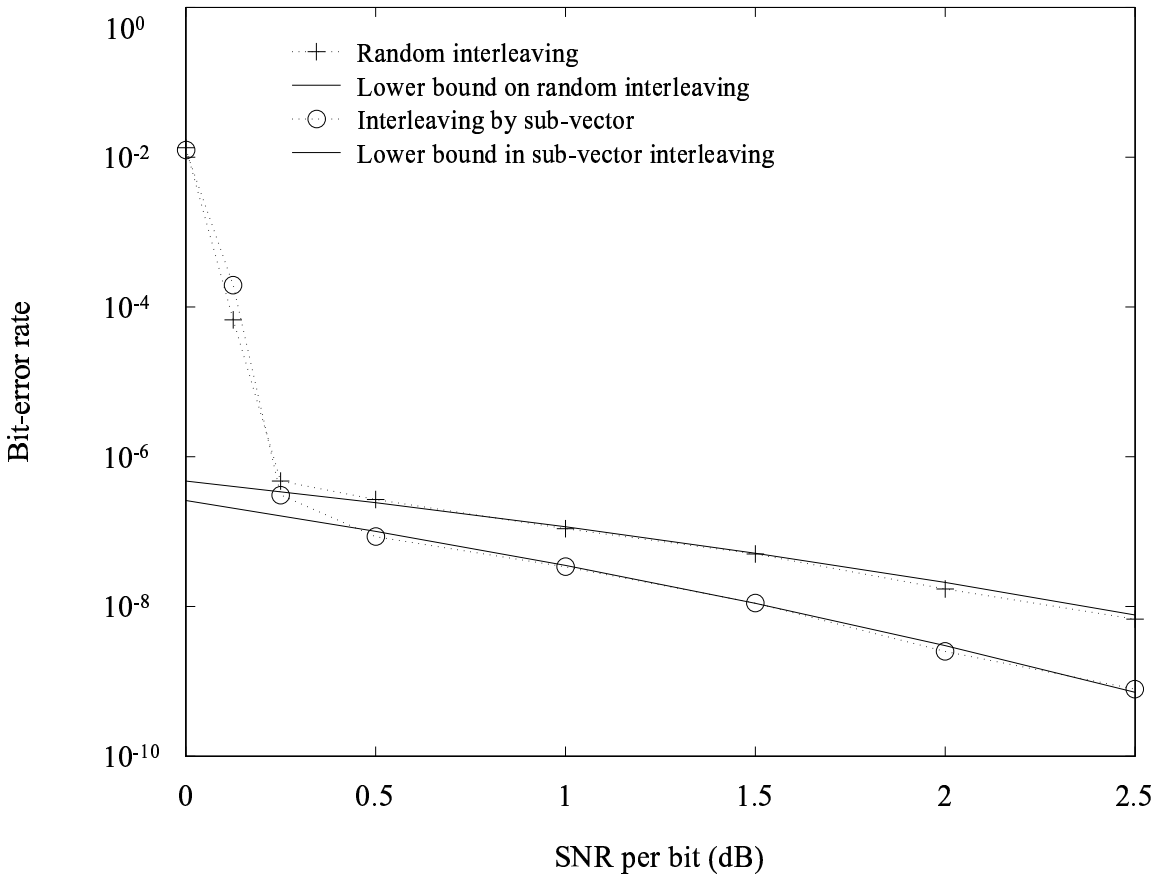

Figure 11: Simulated BER plots for a PCCC employing encoders shown in Fig. 7 and a random interleaver and the same scheme employing an optimal sub-vector interleaver. The interleaver length is 14000 in both cases and the simulated BERs are plotted along with their lower bounds.

3.5 Summary

PCCCs employing sub-vector interleavers have BERs significantly lower than the same schemes employing random interleavers. Sub-vector interleavers are optimal in the sense that they interleave the maximum fraction of terminating input sequences into non-terminating input sequences, as set forth in Lemma 2. At the same time, these interleavers can be designed to

- significantly decrease the BER of the coding scheme at low SNRs by increasing the minimum Hamming weight of codewords with low weight input sequences, and
- increase the free distance of the overall PCCC and thereby lower the BER at high SNRs.

The key feature of sub-vector interleaving is its low-complexity design, which typically involves searching over interleavers equal in length to only a few periods of the feedback polynomial of the encoders. Sub-vector interleaver design yields a type of block interleaver in which columns are permuted in order to eliminate the error patterns that are most significant in terms of their contribution to the BER. By incorporating random interleaving over each of the columns of the block interleaver into the design of sub-vector interleavers, we maintain the gain in performance expected with increased interleaver length. This allows us to scale up our design to form any length-N interleaver without additional computation.

Our simulations show that sub-vector interleavers improve the BER performance of PCCCs by lowering their error floor. This is accomplished by increasing the free distance of the coding scheme and eliminating the lowest Hamming weight codewords generated by weight-2 terminating input sequences. Random interleaving over the columns of the sub-vector interleaver is key to being able to scale this method up to any length interleaver desired without additional design complexity.

The contributions of the research described in this chapter are as follows:

- Dynamical system representation of PCCCs.
- Constructive proofs of Lemma 1 and Lemma 2.
- Sub-vector interleaving for PCCCs, a key method of interleaving that
  - has low complexity design,
  - adds no additional complexity to the encoding or decoding,
  - lowers the error floor of the PCCC,
  - scales up to any length-N interleaver without additional design complexity,
  - increases average free distance codeword weight for the coding scheme employing a sub-vector interleaver with randomly generated row permutations.

CHAPTER IV

INTERLEAVER DESIGN FOR SERIALLY CONCATENATED CONVOLUTIONAL CODES

A typical SCCC, as shown in Fig. 12, comprises an outer, non-systematic convolutional encoder, an interleaver, and an inner, systematic, recursive convolutional encoder connected in series. The input sequence to the outer encoder contains the message bits to be encoded and transmitted. The input sequence to the inner encoder contains an interleaved version of the bits of the codeword generated by the outer encoder.

The discussion in the chapter, previously described in [33], can be applied to an SCCC with either a recursive or non-recursive outer encoder. We base our discussion in the first part of this chapter on an SCCC that uses a rate $\frac{2}{3}$, non-systematic convolutional encoder with total memory $M = 2$ as the outer encoder and a rate $\frac{1}{2}$, eight state (or three delay state) recursive convolutional encoder with generator matrix $G = \left[1, \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$ as the inner encoder, as shown in Fig. 12. This particular SCCC was chosen because the BERs expected from it are fairly high, allowing us to demonstrate the substantial reduction in BER caused by use of our interleavers in a relatively short simulation in Section 4.6. However, it is straightforward to apply the methods presented here to SCCCs employing other types of constituent encoders, and we will show the results of the simulation of a more powerful SCCC in Chapter 5 of this thesis.

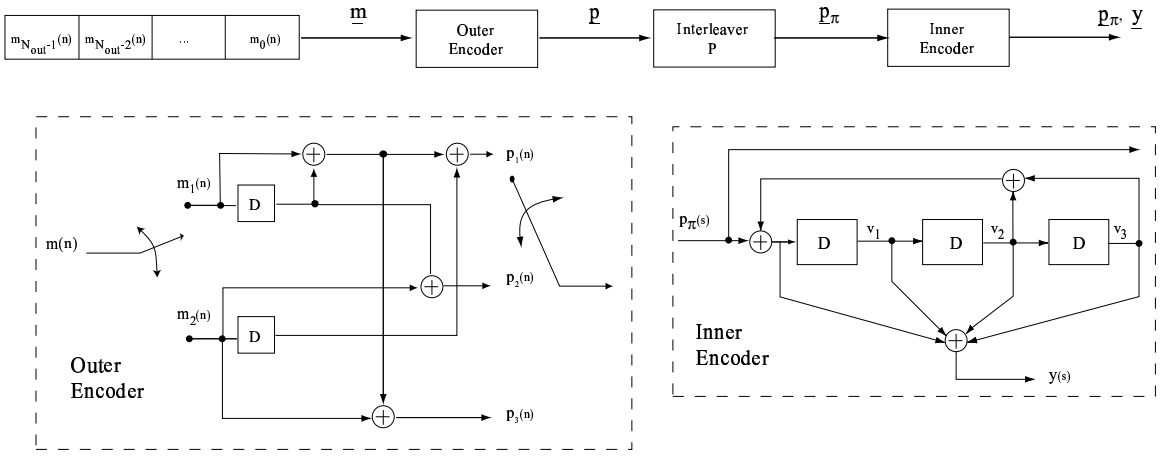

Figure 12: Serially Concatenated Turbo Coding Scheme

4.1 Dynamical System Representation of SCCC

As in the case of interleaver design for PCCCs, our method for designing interleavers for SCCCs is based on the analysis of an SCCC whose constituent encoders are represented by state variable models of discrete dynamical systems.

For the inner encoder shown in Fig. 12, considered in isolation from the rest of the SCCC, the state variable model consists of a state vector $\underline{v}$, containing the delay states v_1, v_2, v_3, and the input $p_\pi(s)$ which is the next bit to be encoded. The output, y, could also be included in the state variable model, but it is not necessary for this discussion. Note that since this is a binary coding scheme all addition is mod 2.

$$\begin{bmatrix} v_1(s+1) \\ v_2(s+1) \\ v_3(s+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_1(s) \\ v_2(s) \\ v_3(s) \end{bmatrix} \oplus p_\pi(s) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

In practice, there are a finite number, N_{in}, of input sequence bits to the inner encoder to be encoded by the inner encoder. We can think of these bits as being placed into a length-N_{in} vector $\underline{p}_\pi$ to be queued into the encoder as shown in Fig. 13. As each bit is shifted to the right into the encoder, we place a 0 into the left-most element of $\underline{p}_\pi$, until all N_{in} input sequence bits have been encoded and $\underline{p}_\pi$ is a vector of zeros. This leads to an interpretation of the encoder as an autonomous dynamical

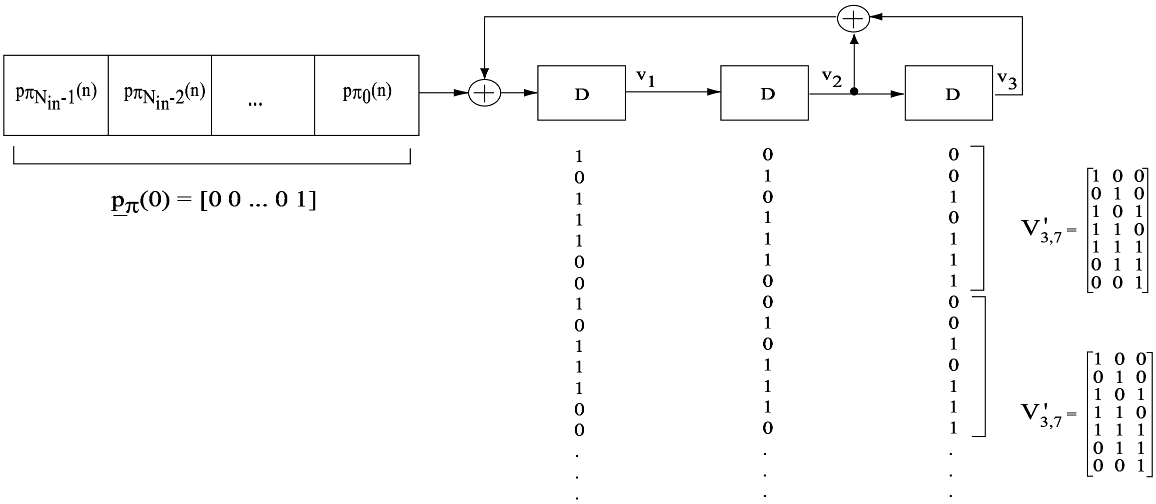

Figure 13: Impulse response of the delay states of the encoder

system where the delay states $\underline{v}$ and input vector $\underline{p}_{\pi}$ are the state variables:

$$\begin{bmatrix} v_1(s+1) \\ v_2(s+1) \\ v_3(s+1) \\ p_{\pi_0}(s+1) \\ \vdots \\ p_{\pi_{N_{in}-3}}(s+1) \\ p_{\pi_{N_{in}-2}}(s+1) \\ p_{\pi_{N_{in}-1}}(s+1) \end{bmatrix} = \begin{bmatrix} 0\;1\;1\;1\;0 \cdots 0\;0 \\ 1\;0\;0\;0\;0 \cdots 0\;0 \\ 0\;1\;0\;0\;0 \cdots 0\;0 \\ 0\;0\;0\;0\;1 \cdots 0\;0 \\ \vdots \quad\quad \ddots \\ 0\;0\;0\;0\;0 \cdots 1\;0 \\ 0\;0\;0\;0\;0 \cdots 0\;1 \\ 0\;0\;0\;0\;0 \cdots 0\;0 \end{bmatrix} \begin{bmatrix} v_1\,(s) \\ v_2\,(s) \\ v_3\,(s) \\ p_{\pi_0}\,(s) \\ \vdots \\ p_{\pi_{N_{in}-3}}\,(s) \\ p_{\pi_{N_{in}-2}}\,(s) \\ p_{\pi_{N_{in}-1}}\,(s) \end{bmatrix} = A \begin{bmatrix} \underline{v}\,(s) \\ \underline{p}_{\pi}\,(s) \end{bmatrix}$$

Furthermore, we notice that after N_{in} time steps the state of the system is given by

$$\begin{bmatrix} \underline{v}\,(N_{in}) \\ \underline{p}_{\pi}\,(N_{in}) \end{bmatrix} = A \cdot \begin{bmatrix} \underline{v}\,(N_{in}-1) \\ \underline{p}_{\pi}\,(N_{in}-1) \end{bmatrix} = A^{N_{in}} \cdot \begin{bmatrix} \underline{v}\,(0) \\ \underline{p}_{\pi}\,(0) \end{bmatrix}$$

where

$$\underline{v}\,(0) = \begin{bmatrix} 0\;0\;0 \end{bmatrix}', \underline{p}_{\pi}\,(N_{in}) = \begin{bmatrix} 0\;0 \cdots 0 \end{bmatrix}',$$

and

$$A^{N_{in}} = \begin{bmatrix} x\;x\;x\;1\;0\;0\;1\;1\;1\;0 \cdots 1\;0\;0\;1\;1\;1\;0 \\ x\;x\;x\;0\;0\;1\;1\;1\;0\;1 \cdots 0\;0\;1\;1\;1\;0\;1 \\ x\;x\;x\;0\;1\;1\;1\;0\;1\;0 \cdots 0\;1\;1\;1\;0\;1\;0 \\ 0\;0\;0\;0\;0\;0\;0\;0\;0\;0 \cdots 0\;0\;0\;0\;0\;0\;0 \\ \vdots \quad\quad\quad\quad\quad \vdots \quad \vdots \quad\quad\quad \vdots \\ 0\;0\;0\;0\;0\;0\;0\;0\;0\;0 \cdots 0\;0\;0\;0\;0\;0\;0 \end{bmatrix}.$$

The first three rows of the matrix $A^{N_{in}}$ from column 4 to column $N_{in}+3$ consist of a pattern of bits that repeats periodically. The remainder of the rows of $A^{N_{in}}$ are filled entirely with zeros. This is true of A^M for all $M \geq N_{in}$. Careful observation shows that these bit patterns, which have the same period as the feedback polynomial of the inner encoder, correspond to what we refer to as the "impulse responses of the delay states of the encoder," as seen in Fig. 13. The rows of A^M containing the delay state impulse responses will change as A^M is iterated; however, since the rows of A^M are linear combinations of the rows of $A^{N_{in}}$, the delay state impulse responses of A^M will always be elements of the vector space spanned by the rows of the delay state impulse responses of $A^{N_{in}}$.

In Fig. 13 we introduce notation for the matrix $V_{k,ln}$ that contains the k delay state impulse responses over n length-l periods of the encoder. The encoder in Fig. 13 has $k=3$ delay states and an impulse response with period $l=7$. Thus, $V_{3,7}$ is a matrix containing the three delay state impulse responses over a single length-7 period of the impulse response of the encoder.

If we remove the interleaver from our coding scheme, we can represent the entire SCCC as a discrete dynamical system. We express the delay states v_1, v_2, and v_3, after every third iteration on the input $p_\pi(s)$ to the inner encoder, corresponding to the three output bits produced after every iteration on the two input bits to the rate $\frac{2}{3}$ outer encoder, as a function of the outputs from the outer encoder, p_1, p_2, and p_3:

$$\begin{bmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{bmatrix} \oplus \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1(n) \\ p_2(n) \\ p_3(n) \end{bmatrix}.$$

Furthermore, we can express the outputs of the outer encoder, p_1, p_2, and p_3, as a function of the input sequence, m:

$$\begin{aligned} p_1(n) &= m(2n) + m(2n-2) + m(2n-3) \\ p_2(n) &= m(2n-1) + m(2n-2) \\ p_3(n) &= m(2n) + m(2n-1) + m(2n-2) \end{aligned}$$

This representation again leads to an interpretation of the entire SCCC as an

autonomous dynamical system where the inner encoder delay states $\underline{v}$, the outer encoder outputs $\underline{p}_o$, and the input sequence vector $\underline{m}$ are the state variables:

$$\begin{bmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \\ p_1(n+1) \\ p_2(n+1) \\ p_3(n+1) \\ m_0(n+1) \\ \vdots \\ m_{N_{out}-3}(n+1) \\ m_{N_{out}-2}(n+1) \\ m_{N_{out}-1}(n+1) \end{bmatrix} = \begin{bmatrix} 1&1&1&1&0&1&0&0&0&0&\cdots&0&0 \\ 1&1&0&0&1&0&0&0&0&0&\cdots&0&0 \\ 0&1&1&1&0&0&0&0&0&0&\cdots&0&0 \\ 0&0&0&0&0&0&1&0&1&1&\cdots&0&0 \\ 0&0&0&0&0&0&0&1&1&0&\cdots&0&0 \\ 0&0&0&0&0&0&1&1&1&0&\cdots&0&0 \\ 0&0&0&0&0&0&0&0&1&0&\cdots&0&0 \\ 0&0&0&0&0&0&0&0&0&1&\cdots&0&0 \\ & & & & & & & & & & \ddots & & \\ 0&0&0&0&0&0&0&0&0&0&\cdots&1&0 \\ 0&0&0&0&0&0&0&0&0&0&\cdots&0&1 \\ 0&0&0&0&0&0&0&0&0&0&\cdots&0&0 \\ 0&0&0&0&0&0&0&0&0&0&\cdots&0&0 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \\ p_1(n) \\ p_2(n) \\ p_3(n) \\ m_0(n) \\ m_1(n) \\ \vdots \\ m_{N_{out}-4}(n) \\ m_{N_{out}-3}(n) \\ m_{N_{out}-2}(n) \\ m_{N_{out}-1}(n) \end{bmatrix} \tag{20}$$

$$= B \begin{bmatrix} \underline{v}(n) \\ \underline{p}_o(n) \\ \underline{m}(n) \end{bmatrix},$$

Note that N_{out}, the length of the input sequence to the outer encoder, is equal to $\frac{2}{3}N_{in}$. Thus, we have

$$\begin{bmatrix} \underline{v}(N_{out}) \\ \underline{p}_o(N_{out}) \\ \underline{m}(N_{out}) \end{bmatrix} = B \cdot \begin{bmatrix} \underline{v}(N_{out}-1) \\ \underline{p}_o(N_{out}-1) \\ \underline{m}(N_{out}-1) \end{bmatrix} = B^{N_{out}} \cdot \begin{bmatrix} \underline{v}(0) \\ \underline{p}_o(0) \\ \underline{m}(0) \end{bmatrix}.$$

Since we assume the outer encoder is in the zero state before the first bits of the input sequence are encoded, we set $\begin{bmatrix} m_0(0) & m_1(0) \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix}$. In this case, the matrix $B^{N_{out}}$ is given by:

$$B^{N_{out}} = \begin{bmatrix} x&x&x&x&x&x&x&x&0&1&1&1&1&1&1&0&0&0&0&1&1&0&\cdots&0&1&1&1&1&1&1&0&0&0&0&1&1&0 \\ x&x&x&x&x&x&x&x&0&1&1&0&0&1&1&1&1&1&1&0&0&0&\cdots&0&1&1&0&0&1&1&1&1&1&1&0&0&0 \\ x&x&x&x&x&x&x&x&1&0&0&0&0&1&1&0&0&1&1&1&1&1&\cdots&1&0&0&0&0&1&1&0&0&1&1&1&1&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&\cdots&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ & & & & & & & & & & & & & & & & & & & & & & \ddots & & & & & & & & & & & & \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&\cdots&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}. \tag{21}$$

We will refer to the first three rows of the matrix $B^{N_{out}}$ from columns 9 to $N_{out}+6$ as the "multiplexed delay state impulse responses," since they correspond, with some

cyclic shifting (see Fig. 13), to two delay state impulse responses multiplexed into a single sequence.

An important relationship exists between the multiplexed delay state impulse responses and a class of input sequences that correspond to terminating codewords. This relationship leads us to a method described in Section 4.2 for terminating both the outer and inner encoder in the zero state by simply appending a tail to the input sequence to the SCCC.

4.2 Terminating the Inner and Outer Encoders

The outer non-recursive convolutional encoder in an SCCC is terminated by an input sequence $\underline{m}_t$, which has a tail sequence of zeros appended to it of length equivalent to the memory of the encoder (i.e. $\underline{m}_t = \left[\underline{m}\ 0\ 0\right]$). The inner encoder in an SCCC is terminated by an input sequence $\underline{p}_{\pi_t}$ that begins with the encoder in the all-zero state (i.e., with $\underline{p}_o(0) = \left[0\ 0\ 0\right]'$ and $\underline{v}(0) = \left[0\ 0\ 0\right]'$) and returns the encoder to the all-zero state after its last non-zero bit has been encoded. Since the inner encoder of an SCCC is typically recursive, most codewords generated by the outer encoder are interleaved into input sequences $\underline{p}_\pi$ that do not terminate the inner encoder. As for PCCCs, we increase the weight of the parity sequences associated with non-terminating error patterns that appear at the end of the input sequences to the encoders by appending tail sequences that terminate the constituent encoders. In this section, we will show that it is possible to terminate *both* encoders in the zero state by appending a short tail to the *input sequence* $\underline{m}$, which permits the type of interleaving that we will discuss in Section 4.3. We will show in Section 4.3 that these interleavers:

- *Increase the interleaver gain* of the SCCC and
- Can be scaled up to any desired length without additional design complexity.

By definition, a terminating input sequence to the inner encoder will leave the inner encoder in the zero-state after all of its bits have been encoded. After a terminating input sequence $\underline{p}_{\pi_t}$ has been encoded, the state of the inner encoder is given by $\underline{v}\left(N_{out}\right) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}'$. Simplifying (20) through (21) and incorporating the notation for $V_{3,14}$, which is a 3×14 matrix with dimensions chosen so that each row contains exactly one period of each of the multiplexed delay state impulse responses, we get the following expression for the state of the inner encoder after the input sequence $\underline{m}$ to the outer encoder that generates the input sequence $\underline{p}_t$ to the inner encoder has been encoded:

$$\begin{aligned} \begin{bmatrix} v_1\left(N_{out}\right) \\ v_2\left(N_{out}\right) \\ v_3\left(N_{out}\right) \end{bmatrix} &= \begin{bmatrix} V_{3,14} \cdots V_{3,14} \end{bmatrix} \begin{bmatrix} m\left(0\right) \\ \vdots \\ m\left(N_{out}-1\right) \end{bmatrix} = \begin{bmatrix} V_{3,14} \cdots V_{3,14} \end{bmatrix} \cdot \underline{m} \\ &= \begin{bmatrix} V_{3,14} \cdots V_{3,14} \end{bmatrix} \cdot \begin{bmatrix} \underline{m}_1 \\ \vdots \\ \underline{m}_{\frac{N_{out}}{14}} \end{bmatrix} \\ &= \left[V_{3,14}\right] \cdot \left[\underline{m}_1 \oplus \underline{m}_{t_2} \oplus \cdots \oplus \underline{m}_{\frac{N_{out}}{14}}\right] = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \end{aligned} \tag{22}$$

From (22), we see that a sufficient condition to guarantee that a particular input sequence $\underline{m}$ generates a codeword $\underline{p}_t$ that terminates the inner encoder would be for each row of the $14 \times \frac{N_{out}}{14}$ matrix composed of input sequence bits to the outer encoder read in column-wise to have a parity sum equal to zero. It is apparent that this condition can be achieved by appending the appropriate length-14 tail sequence to the end of our original input sequence $\underline{m}$. Careful analysis of individual constituent encoders reveals that only 3 bits appended to $\underline{m}$ are required to terminate the inner encoder. This result is derived in Section 4.5.

By setting the last two bits of the input sequence to the outer encoder $\underline{m}$ equal to zero (i.e., $\begin{bmatrix} m\left(N_{out}-2\right) & m\left(N_{out}-1\right) \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix}$), we guarantee that the outer encoder is always terminated in the zero state. Since this does not affect the parity sum in (22), we now have both the inner and outer encoders in the SCCC terminated by a tail sequence appended to the input sequence to the outer encoder.

This type of termination is significant because it can be achieved by appending a tail to the end of the input sequence to the *outer* encoder, not the inner encoder. This result, achieved with insight gained from the dynamical system analysis of the SCCC, has not been shown elsewhere, and it makes possible the increase in interleaver gain achieved by the method of interleaving that is described in Section 4.3.

4.3 Sub-vector Interleaving

The goal of the interleaver, $P_{N_{in}}$, for an SCCC[1]is to permute all low weight output sequences from the outer encoder, $\underline{p}$, into input sequences, $\underline{p}_{\pi}$, that do not terminate the inner encoder. If we restrict the set of valid input sequences to the outer encoder to contain only those sequences that terminate both encoders, then a maximum likelihood decoding algorithm would correct error patterns that do not correspond to sequences that terminate both encoders. Thus, an interleaver that permutes low weight codewords from the outer encoder into non-terminating input sequences to the inner encoder would prevent the most significant set of error patterns, in terms of their BER contribution, from being decoded as codewords of the SCCC.

In an expression similar to (22), we can relate an un-interleaved input sequence that terminates the inner encoder, $\underline{p}_t$, to the inner encoder delay state impulse responses:

$$\left[V_{3,7} \cdots V_{3,7} \right] \cdot \underline{p}_t = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{23}$$

The notation $V_{3,7}$ describes a 3×7 matrix with dimensions chosen so that each row contains exactly one period of each of the delay state impulse responses of the inner encoder. We would like for our interleaver, $P_{N_{in}}$, to permute the bits of the output sequence from the outer encoder such that the low weight codewords from the outer encoder, $\underline{p}$, do not form terminating input sequences for the inner encoder:

[1]We will use notation $P_{N_{in}}$ to describe an interleaver that permutes the output sequence from the outer encoder of an SCCC into the length-N_{in} input sequence to the inner encoder.

$$\left[V_{3,7} \cdots V_{3,7} \right] \cdot P_{N_{in}} \cdot \underline{p}_t = \left[V_{3,7} \cdots V_{3,7} \right] \cdot \underline{p}_\pi \neq \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Expressing an input sequence $\underline{p}_t$ that satisfies (23) as the vertical concatenation of length-7 sub-vectors, we see that the (mod 2) summation of these sub-vectors is also a terminating input sequence:

$$\begin{aligned} \left[V_{3,7} \cdots V_{3,7} \right] \cdot \underline{p}_t &= \left[V_{3,7} \cdots V_{3,7} \right] \cdot \begin{bmatrix} \underline{p}_{t1} \\ \vdots \\ \underline{p}_{t\frac{N}{7}} \end{bmatrix} \\ &= [V_{3,7}] \cdot \left[\underline{p}_{t1} \oplus \underline{p}_{t2} \oplus \cdots \oplus \underline{p}_{t\frac{N}{7}} \right] \\ &= [V_{3,7}] \cdot \underline{p}_{t_{sub}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \end{aligned}$$

By searching over all length-7 interleavers, it is often possible to find an interleaver P_7 that permutes the weight-d_{free} and weight-$(d_{free}+1)$ input sequences $\underline{p}_{t_{sub}}$ (and all of their cyclic shifts over a length-7 sub-vector) into input sequences that do not correspond to codewords for the inner encoder:

$$[V_{3,7}] \cdot P_7 \cdot \underline{p}_{t_{sub}} \neq \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Since the dimensions of P_7 correspond to the period of the feedback polynomial of our recursive convolutional encoder, we can place P_7 on the diagonal of an $N_{in} \times N_{in}$ (choosing N_{in} to be a multiple of 7) permutation matrix to create a length-N_{in} interleaver $P_{N_{in}}$:

$$\begin{aligned} &\left[V_{3,7} \cdots V_{3,7} \right] \cdot \underbrace{\begin{bmatrix} P_7 & 0 & \cdots & 0 \\ 0 & P_7 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & P_7 \end{bmatrix}}_{P_{N_{in}}} \cdot \begin{bmatrix} \underline{p}_{\pi 1} \\ \vdots \\ \underline{p}_{\pi \frac{N_{in}}{7}} \end{bmatrix} \\ &= [V_{3,7}] \cdot P_7 \cdot \left[\underline{p}_{\pi 1} \oplus \underline{p}_{\pi 2} \oplus \cdots \oplus \underline{p}_{\pi \frac{N_{in}}{7}} \right]. \end{aligned}$$

The interleaver $P_{N_{in}}$ shares the property of P_7 in that it interleaves all shifts of the weight-d_{free} and weight-$d_{free}+1$ input sequences corresponding to codewords for

the outer encoder into input sequences that do not correspond to a codeword for the inner encoder. We refer to this method of interleaving as "sub-vector interleaving" for SCCCs.

It is straightforward to show that, by adding the type of tail described in Section 4.2 to the end of our input sequence, the length-7 sub-vectors of the output sequence, $\underline{p} = \left[\underline{p}_1 \ \cdots \ \underline{p}_{\frac{N_{in}}{7}} \right]'$, of the outer encoder have the property

$$\left[\underline{p}_1 \oplus \underline{p}_2 \oplus \cdots \oplus \underline{p}_{\frac{N_{in}}{7}} \right] = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Since interleaving over a length-7 sub-vector effectively permutes the columns of a $\frac{N_{in}}{7} \times 7$ matrix with the sequence $\underline{p}$ read in row-wise, we have that $\underline{p}_\pi$ also satisfies

$$\left[\underline{p}_{\pi 1} \oplus \underline{p}_{\pi 2} \oplus \cdots \oplus \underline{p}_{\pi \frac{N_{in}}{7}} \right] = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Therefore, the interleaved input sequence $\underline{p}_\pi$ also terminates the inner encoder in the zero state since

$$\left[V_{3,7} \ \cdots \ V_{3,7} \right] \cdot \underline{p}_\pi = V_{3,7} \cdot \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

The sub-vector interleaver $P_{N_{in}}$ effectively permutes the rows of a $\frac{N_{in}}{7} \times 7$ matrix with information bits read in row-wise. Since some weight-d, where $d \geq (d_{free} + 2)$, input sequences to the inner encoder corresponding to codewords of the outer encoder will be interleaved into terminating input sequences for the inner encoder, we need to increase the expected weight of the parity sequences from the inner encoder of these types of input sequences by employing some type of interleaving over each of the columns of the sub-vector interleaver. As in sub-vector interleaver design for PCCCs, the simplest approach is to interleave randomly over each column of the sub-vector interleaver. This approach has the unique advantage of being practical for use in the design of extremely long interleavers.

In Section 4.6 we will show the results of a Monte Carlo simulation of an SCCC using our method of sub-vector interleaving integrated with random row interleaving to create a length-231 sub-vector interleaver.

4.4 Increasing Interleaver Gain in an SCCC

A sub-vector interleaver designed to interleave all weight-d_{free} and weight-$(d_{free}+1)$ codewords from the outer encoder into non-terminating input sequences for the inner encoder would prevent error patterns corresponding to these sequences from being incorrectly decoded as codewords of the SCCC, given maximum likelihood decoding. Similarly to what was done in Section (2.4.1.2), we compute the probability of a decoding error resulting from an odd weight-$(d_{free}+2)$ error pattern corresponding to a codeword from the outer encoder that is interleaved into an input sequence that terminates the inner encoder:

$$P_b\left(e|\underline{p}_{d_{free}+2,p}\right) \approx \left(\frac{w}{N_{out}}\right)\frac{\left(N^o_{(d_{free}+2)}\right)\begin{pmatrix} N_{in} \\ \frac{(d_{free}+1)}{2} \end{pmatrix}}{\begin{pmatrix} N_{in} \\ d_{free}+2 \end{pmatrix}} Q\left(\sqrt{(d_{free}+2+p)\cdot\frac{2R_c\varepsilon_b}{N_0}}\right) \tag{24}$$
$$\propto (N_{in})^{-\frac{(d_{free}+3)}{2}} \cdot e^{-(d_{free}+2+p)\left(\frac{R_c\varepsilon_b}{N_0}\right)},$$

where w is the weight of the input sequence to the outer encoder, N_{out} is the length of the input sequence to the outer encoder, $N^o_{(d_{free}+2)}$ is the number of weight-$(d_{free}+2)$ codewords from the outer encoder, l is the distance from the first to last non-zero bit of the interleaved error pattern, $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{\frac{-t^2}{2}}dt$, R_c is the overall rate of the SCCC, and $\frac{\varepsilon_b}{N_0}$ is the SNR per bit.

Comparing the BER contribution of the most significant error pattern in a sub-vector interleaved SCCC calculated in (24) to the BER contribution of the most significant error pattern in an SCCC employing an average random interleaver,

$$P_b\left(e|\underline{p}_{d_{free},p}\right) \propto (N_{in})^{-\frac{(d_{free}+1)}{2}} \cdot e^{-(d_{free}+p)\left(\frac{R_c\varepsilon_b}{N_0}\right)}, \tag{25}$$

shows that we have increased the interleaver gain of the sub-vector interleaved SCCC by a factor of N_{in}.

4.4.1 Upper Bound on Interleaver Gain

We have shown that the interleaver gain in an SCCC can be increased by using sub-vector interleaving to increase the minimum weight of terminating input sequences to the inner encoder. Since sub-vector interleaving does not address the interleaving of multiple terminated output sequences from the outer encoder into non-terminating input sequences for the inner encoder, a loose upper bound on the interleaver gain exists for sub-vector interleaving.

Fig. 14 shows how two weight-d_{free} terminated output sequences from the outer encoder are interleaved into d_{free} weight-2 terminating input sequences for an SCCC shown in Fig. 15 with identical eight state recursive convolutional outer and inner encoders.

We would like to use the notation

$$\underline{\pi}_{14} = \left[\pi_0 \ \pi_1 \ \cdots \ \pi_{12} \ \pi_{13} \right]$$

to describe an interleaver that operates on a length-14 sequence where the i^{th} element of the interleaved sequence is the π_i^{th} element of the input sequence. In this way, the column permutation used by the sub-vector interleaver in the example in Fig. 14 is given by

$$\underline{\pi}_{14} = \left[8 \ 2 \ 5 \ 12 \ 6 \ 9 \ 0 \ 4 \ 11 \ 7 \ 3 \ 10 \ 1 \ 3 \right].$$

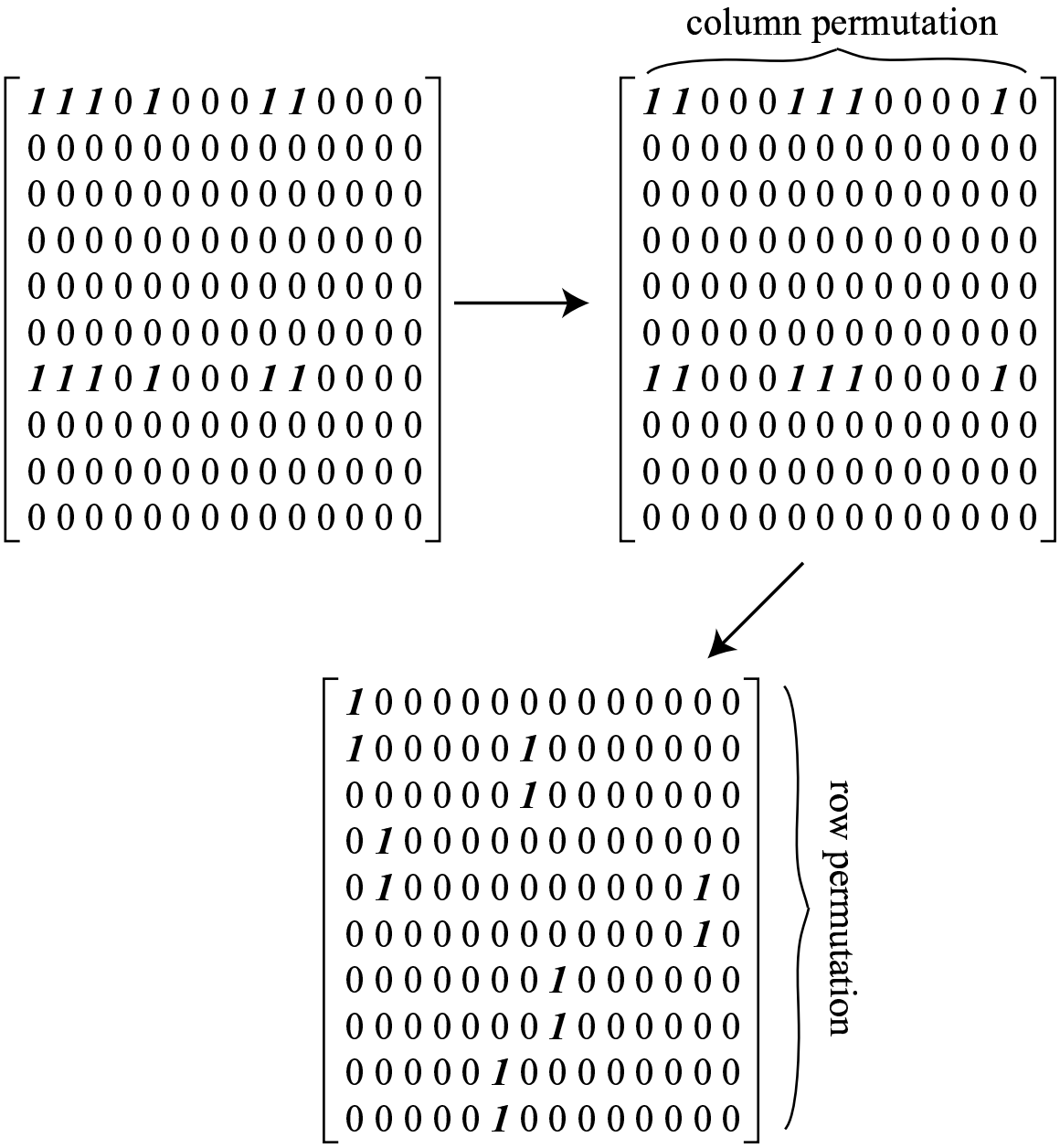

Figure 14: Two weight-d_{free} terminated output sequences interleaved into d_{free} weight-2 terminating input sequences.

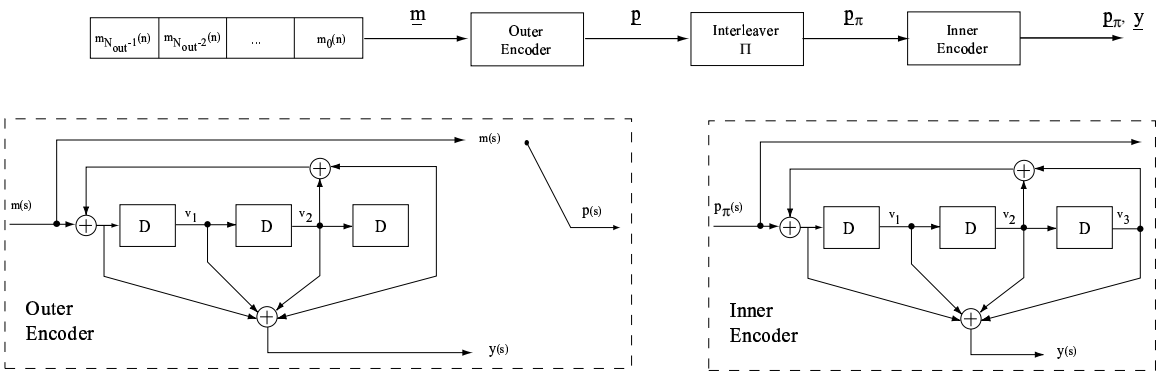

Figure 15: A diagram of an SCCC employing identical rate-$\frac{1}{2}$ recursive convolutional encoders with three delay states.

The probability of a bit error resulting from the incorrect decoding of two weight-d_{free} codewords that have been interleaved into d_{free} weight-2 terminating input sequences for the inner encoder is computed as follows:

$$\begin{aligned} P_b\left(e|\underline{p}_{2d_{free},p}\right) &\approx \left(\tfrac{w}{N_{out}}\right)\frac{\binom{N_{out}}{2}\binom{N_{in}}{d_{free}}}{\binom{N_{in}}{2d_{free}}}Q\left(\sqrt{(2d_{free}+p)\left(\tfrac{2R_c\varepsilon_b}{N_0}\right)}\right) \\ &\propto \tfrac{N_{out}}{{N_{in}}^{d_{free}}}e^{-\left(2d_{free}+p\right)\left(\frac{R_c\varepsilon_b}{N_0}\right)} = \tfrac{1}{2{N_{in}}^{d_{free}-1}}e^{-\left(2d_{free}+p\right)\left(\frac{R_c\varepsilon_b}{N_0}\right)} \end{aligned} \tag{26}$$

In (26), we see that the interleaver gain of the SCCC can be increased to at most $\frac{1}{{N_{in}}^{d_{free}-1}}$ by sub-vector interleaving.[2] Therefore, sub-vector interleavers need not be designed to interleave greater than weight-$(2d_{free}-2)$ into non-terminating input sequences for the inner encoder for the purpose of increasing interleaver gain.

A tighter upper bound on interleaver gain exists and can be seen by computing an upper bound on the minimum weight of a single terminating input sequence to the inner encoder that must exist given any type of sub-vector interleaver that leaves the inner and outer encoder terminated in the zero state. No matter what type of outer encoder the SCCC employs, the terminated output sequences from the outer

[2]Note that for $d_{free}=3$, the interleaver gain is not increased by sub-vector interleaving since $\frac{(d_{free}+1)}{2} = d_{free}-1$. However, the error floor can still be lowered in this case by sub-vector interleaving.

encoder constitute a dimension m vector space Γ_p. If the outer encoder is a systematic recursive convolutional encoder, the terminated output sequences from the outer encoder comprise a $N_{out} - k_o$ dimension vector space, where N_{out} is the length of the input sequence to the outer encoder, and k_o is the number of delay states in the outer encoder. Interleaved versions of the terminated output sequences from the outer encoder form the input sequences to the inner encoder. These input sequences, $\underline{p}_\pi$, are linear combinations of the interleaved versions of the basis vectors, $\{b_{\pi_1}, b_{\pi_2}, \ldots, b_{\pi_m}\}$, which span the dimension m vector space Γ_{p_π} constituted by the 2^m interleaved terminated output sequences from the outer encoder:

$$\underline{p}_\pi = \underline{a} \begin{bmatrix} \underline{b}_{\pi_1} \\ \underline{b}_{\pi_2} \\ \vdots \\ \underline{b}_{\pi_{m-1}} \\ \underline{b}_{\pi_m} \end{bmatrix},$$

where $\underline{a} = \begin{bmatrix} a_1 \; a_2 \; \cdots \; a_{m-1} \; a_m \end{bmatrix}$ is an arbitrary $1 \times m$ binary vector.

The final state of the inner encoder is a linear mapping, L, of the interleaved output sequence from the outer encoder to a dimension k_i vector space Γ_{v_i}, where k_i corresponds to the number of delay states of the inner recursive convolutional encoder:

$$L : \underline{p}_\pi \in \Gamma_{p_\pi} \rightarrow \underline{v}_i \in \Gamma_{v_i},$$

where $\underline{v}_i$ is a length-k_i vector containing the values of the k_i delay states of the inner encoder after the length-N_{in} vector $\underline{p}_\pi$ is encoded. Since L is a linear mapping,

$$L\left(\underline{p}_\pi\right) = a_1 L\left(\underline{b}_{\pi_1}\right) + a_2 L\left(\underline{b}_{\pi_2}\right) + \cdots + a_m L\left(\underline{b}_{\pi_m}\right).$$

We have assumed that the sub-vector interleaver is designed to leave both the inner and outer encoder terminated. Therefore, valid input sequences to the inner encoder (i.e., terminating input sequences) satisfy the following:

$$L\left(\underline{p}_\pi\right) = a_1 L\left(\underline{b}_{\pi_1}\right) + a_2 L\left(\underline{b}_{\pi_2}\right) + \cdots + a_m L\left(\underline{b}_{\pi_m}\right) = \begin{bmatrix} 0 \cdots 0 \end{bmatrix}. \tag{27}$$

From (27) we see that valid input sequences are mapped by the inner encoder to linearly dependent combinations of the vectors $\underline{v}_i \in V_i$.

Lemma 3 *The minimum weight codeword from the outer encoder that corresponds to an input sequence to the outer encoder that leaves both the outer and inner encoders in an SCCC terminated in the zero state is bounded from above by the minimum weight of all codewords* $\underline{p}_{max}$*, where* $\underline{p}_{max}$ *is the maximum weight codeword in the vector subspace of* Γ_p *spanned by the* k_{i+1} *basis vectors* $\left\{\underline{b}_{\pi_{s_1}}, \underline{b}_{\pi_{s_2}}, \ldots, \underline{b}_{\pi_{s_{k_i}}}, \underline{b}_{\pi_{s_{k_i+1}}}\right\}$*, for all* $\binom{m}{k_i+1}$ *combinations of* $\left\{\underline{b}_{\pi_{s_1}}, \underline{b}_{\pi_{s_2}}, \ldots, \underline{b}_{\pi_{s_{k_i}}}, \underline{b}_{\pi_{s_{k_i+1}}}\right\}$ *such that* $s_1, s_2, \ldots, s_{k_i}, s_{k_i+1} \leq m$ *and* $s_1 \neq s_2 \neq \cdots \neq s_{k_i} \neq s_{k_i+1}$.

Proof. The inner encoder is a linear mapping the vector subspace of Γ_{p_π} spanned by the interleaved basis vectors $\left\{\underline{b}_{\pi_{s_1}}, \underline{b}_{\pi_{s_2}}, \ldots, \underline{b}_{\pi_{s_{k_i}}}, \underline{b}_{\pi_{s_{k_i+1}}}\right\}$ to the k_i dimension vector space Γ_{v_i}. Since $\left\{\underline{b}_{\pi_{s_1}}, \underline{b}_{\pi_{s_2}}, \ldots, \underline{b}_{\pi_{s_{k_i}}}, \underline{b}_{\pi_{s_{k_i+1}}}\right\}$ spans a dimension $k_i + 1$ vector space, there is a linear dependence in the vectors corresponding to their mapping to a k_i dimension vector space

$$\Gamma_{v_i} : \left\{L\left(\underline{b}_{\pi_{s_1}}\right), L\left(\underline{b}_{\pi_{s_2}}\right), \ldots, L\left(\underline{b}_{\pi_{s_{k_i}}}\right), L\left(\underline{b}_{\pi_{s_{k_i+1}}}\right)\right\}.$$

The linearly dependent combinations of the vectors in the space spanned by

$$\left\{L\left(\underline{b}_{\pi_{s_1}}\right), L\left(\underline{b}_{\pi_{s_2}}\right), \ldots, L\left(\underline{b}_{\pi_{s_{k_i}}}\right), L\left(\underline{b}_{\pi_{s_{k_i+1}}}\right)\right\}$$

correspond to terminated codewords from the outer encoder that are interleaved into terminating input sequences to the inner encoder. The codewords that are mapped to linearly dependent vectors in Γ_{v_i} are determined by the particular interleaver employed. Therefore, an upper bound exists on the minimum weight of terminating input sequences to the inner encoder that correspond to terminated output sequences from the outer encoder of a particular SCCC employing an arbitrary sub-vector interleaver. This upper bound is the minimum weight of all vectors $\underline{p}_{max}$, where $\underline{p}_{max}$ is the maximum weight codeword in a $k_i + 1$ dimension subspace of Γ_{p_π}. ■

Lemma 4 *Codewords that correspond to vectors in Γ_{p_π} that are mapped to zero vectors in Γ_{v_i} form a vector subspace of Γ_{p_π}.*

Proof. Since $L(\cdots)$ is a linear transformation, if $L\left(\underline{p}_{\pi_1}\right) = \left[0 \cdots 0\right]$ and $L\left(\underline{p}_{\pi_2}\right) = \left[0 \cdots 0\right]$, then $L\left(\underline{p}_{\pi_1}\right) + L\left(\underline{p}_{\pi_2}\right) = L\left(\underline{p}_{\pi_1} + \underline{p}_{\pi_2}\right) = \left[0 \cdots 0\right]$. Therefore, the vectors $\underline{p}_\pi \in \Gamma_{p_\pi}$ such that $L\left(\underline{p}_\pi\right) = \left[0 \cdots 0\right]$ form a vector subspace of Γ_{p_π}. ■

A basis of a $k_i + \gamma$ dimension subspace $\Gamma_{k_i+\gamma} \subseteq \Gamma_p$ will be interleaved and mapped by the inner encoder to Γ_{v_i}. This basis will be mapped to at least γ linearly dependent vectors in Γ_{v_i}. Thus, the vectors of the $k_i + \gamma$ dimension subspace of Γ_p that are interleaved and mapped to the zero vector in Γ_{v_i} constitute a γ, or higher, dimension subspace $\Gamma_\gamma \subseteq \Gamma_{k_i+\gamma}$. It is well-known that the minimum weight codeword in an (n, γ) binary block code is less than or equal to $\frac{n2^{\gamma-1}}{2^\gamma - 1}$ for $\gamma > 1$. We calculate that the maximum dimension of Γ_γ as

$$\gamma_{max} = R(n - l) - k_i + 1,$$

where l is the minimum length of a terminated output sequence from the outer encoder, $n > l$ is the length of the block code, and R is the rate of the outer encoder. Since the codewords in Γ_γ form a binary block code, we have another upper bound on the maximum weight of a terminating input sequence to the inner encoder that corresponds to a terminated output sequence from the outer encoder.

Lemma 5 *The minimum weight codeword from the outer encoder that corresponds to an input sequence to the outer encoder that leaves both the outer and inner encoders in an SCCC terminated in the zero state is bounded from above by the minimum value of $\frac{n2^{\gamma_{max}-1}}{2^{\gamma_{\max}}-1}$ for all n equal to an integer multiple of $\frac{1}{R}$.*

The function $f(n) = \frac{n2^{\gamma_{max}-1}}{2^{\gamma_{\max}}-1}$ for n equal integer multiples of $\frac{1}{R}$ is a local minimum when

$$f(n) < f\left(n + \frac{1}{R}\right) \text{ and } f(n) < f\left(n - \frac{1}{R}\right).$$

This occurs when both

$$\log_2(nR+2) < R(n-l) - k_i + 2$$

and

$$R(n-l) - k_i + 1 < \log_2(nR+1).$$

The results of this minimization for different SCCC configurations are given in Table (1). These values are compared to the achievable minimum weight terminating input sequence weight determined experimentally by searching over all possible sub-vector interleavers matched to a particular SCCC.

The following example will illustrate the concepts described in Lemmas 3-5 for the SCCC in Fig. 12.

Example 2 *For the SCCC in Fig. 15 with identical eight state outer and inner recursive convolutional encoders, a basis for a five dimensional vector subspace spanned by terminated codewords from the outer encoder is given by:*

$$\begin{bmatrix} \underline{b}_1 \\ \underline{b}_2 \\ \underline{b}_3 \\ \underline{b}_4 \\ \underline{b}_5 \end{bmatrix} = \begin{bmatrix} 1\,1\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,1\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,1\,1\,1\,1 \end{bmatrix}.$$

Since these five basis vectors will be mapped by the interleaver and inner encoder to a three dimensional vector space, Γ_{v_i}, we know that there at least two vectors corresponding to the mapping of

$$\{\underline{b}_1, \underline{b}_2, \underline{b}_3, \underline{b}_4, \underline{b}_5\}$$

to

$$\Gamma_{v_i} : \{L(\underline{b}_1), L(\underline{b}_2), L(\underline{b}_3), L(\underline{b}_4), L(\underline{b}_5)\}$$

Table 1: A table relating outer encoder free distance codeword weights to both the upperbound on and the actual minimum weight terminating input sequences to the inner encoder for SCCCs employing sub-vector interleavers and different combinations of inner and outer encoders.

Outer Encoder		*Inner Encoder*		
Generator Matrix $G =$	$d_{free} =$	*Generator Matrix* $G =$	*Minimum Terminating Input Sequence Weight*	
			Upper Bound	*Achievable*
$\left[1,\ \frac{\left(1+D+D^2\right)}{(1+D^2)}\right]$	5	$\left[1,\ \frac{\left(1+D+D^2\right)}{(1+D^2)}\right]$	8	7
$\left[1,\ \frac{\left(1+D+D^2\right)}{(1+D^2)}\right]$	5	$\left[1,\ \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$	9	7
$\left[1,\ \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$	6	$\left[1,\ \frac{\left(1+D+D^2\right)}{(1+D^2)}\right]$	9	7
$\left[1,\ \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$	6	$\left[1,\ \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$	10	8

are linearly dependent on the others.

The weight enumerator of codewords in the vector space spanned by $\{\underline{b}_1, \underline{b}_2, \underline{b}_3, \underline{b}_4, \underline{b}_5\}$ *is given by*

$$A(z) = 1 + 4z^6 + 10z^7 + 6z^8 + 4z^9 + 3z^{10} + 2z^{11} + z^{12} + z^{14}.$$

Because of the linear dependence in the vectors $\{L(\underline{b}_1), L(\underline{b}_2), L(\underline{b}_3), L(\underline{b}_4), L(\underline{b}_5)\}$*, we know that at least four codewords (including the zero codeword) in the span of* $\{\underline{b}_1, \underline{b}_2, \underline{b}_3, \underline{b}_4, \underline{b}_5\}$ *are interleaved in to a terminating input sequence for the inner encoder. Therefore, by considering the weight enumerator for the codebook spanned by* $\{\underline{b}_1, \underline{b}_2, \underline{b}_3, \underline{b}_4, \underline{b}_5\}$*, we gave found that an upper bound on the minimum weight codeword to terminate the inner encoder is* 11*.*

The codewords in the span of $\{\underline{b}_1, \underline{b}_2, \underline{b}_3, \underline{b}_4, \underline{b}_5\}$ *that are interleaved into terminating input sequences for the inner encoder form a two (or greater) dimensional vector subspace of the vector space spanned by* $\{L(\underline{b}_1), L(\underline{b}_2), L(\underline{b}_3), L(\underline{b}_4), L(\underline{b}_5)\}$*. We know that the average weight of codewords in this vector space is equal to*

$$\frac{n2^{\gamma_{max}-1}}{2^{\gamma_{\max}}-1} \leq \frac{16 \cdot 2^{2-1}}{2^2-1} = 10\frac{2}{3}.$$

The minimum weight of codewords in this vector space cannot be greater than the average weight; therefore, a tighter upper bound on the minimum weight codeword interleaved into a terminating input sequence for the inner encoder is 10*. In practice, we have found that the true upper bound on the minimum weight terminating input sequence to the inner encoder is* 8*, because there does not exist a two-dimensional vector subspace of* $\{\underline{b}_1, \underline{b}_2, \underline{b}_3, \underline{b}_4, \underline{b}_5\}$ *that does not include a weight-*8 *codeword.*

4.5 Trellis Termination with a Minimum Length Tail Sequence

In Section 4.2, we showed that it is possible to terminate both the inner and outer encoders in an SCCC with a single tail sequence appended to the input sequence

Table 2: A table of outer encoder terminating input sequences and terminated output sequences that generate weight-6 and weight-7 input sequences to the inner encoder for the SCCC in Fig. 15.

Input Sequence, $\underline{m}$		Output Sequence, $\underline{y}$
$[1\ 0\ 1\ 1\ 0\ 0\ 0]$	$\rightarrow$	$[1\ 1\ 1\ 1\ 0\ 0\ 0]$
$[1\ 1\ 1\ 0\ 1\ 0\ 0]$	$\rightarrow$	$[1\ 0\ 0\ 0\ 1\ 0\ 0]$
$[1\ 1\ 0\ 0\ 0\ 1\ 0]$	$\rightarrow$	$[1\ 0\ 1\ 1\ 0\ 1\ 0]$
$[1\ 0\ 0\ 0\ 1\ 0\ 1]$	$\rightarrow$	$[1\ 1\ 0\ 1\ 0\ 0\ 1]$

to the outer encoder. This type of termination allows for the method of sub-vector interleaving described in Section 4.3.

In this section we will describe the process of designing a minimum length terminating tail sequence and illustrate the process with the use of a detailed example. We will show that such a tail sequence must be no longer than the combined length of the sequences necessary to terminate the encoders separately.

The SCCC for which we will design a minimum length terminating tail sequence employs two identical rate-$\frac{1}{2}$ recursive convolutional encoders with three delay states and generator matrix $G = \left[1,\ \frac{\left(1+D+D^2+D^3\right)}{(1+D^2+D^3)}\right]$. A diagram of this SCCC is shown in Fig. 15.

4.5.1 Sub-vector Interleaver Choice

A sub-vector interleaver that interleaves the low-weight terminated codewords from the outer encoder into non-terminating input sequences for the inner encoder should be chosen for use in an SCCC. The lowest weight codewords generated by the outer encoder in the SCCC in Fig. 15 have weight = 6 and weight = 7. These codewords form the input sequences to the inner encoder listed in Table 7. The inner encoder terminating input sequences and the terminated output sequences that correspond to the low weight input sequences to the inner encoder are listed in Table 6.

Table 3: A table of the weight-6 and weight-7 input sequences to the inner encoder that correspond to terminated codewords generated by the outer encoder of the SCCC in Fig. 15.

Input Sequence, $\underline{p}$
$[1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0]$
$[1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0]$
$[1\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0]$
$[1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 1]$

Table 4: A table of the weight-6 and weight-7 permuted input sequences to the inner encoder that correspond to terminated codewords generated by the outer encoder of the SCCC in Fig. 15.

Permuted Input Sequence, $\underline{p}_\pi$
$[1\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0]$
$[1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 0]$
$[1\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1]$
$[1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0]$

The terminating input sequence

$$\underline{m} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the terminated output sequence

$$\underline{y} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

for the outer encoder generate a terminating input sequence

$$\underline{p} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

for the inner encoder. Therefore, it is necessary to design a sub-vector interleaver to interleave the input sequence to the inner encoder so that there are no weight-6 or weight-7 terminating input sequences that correspond to terminated codewords from the outer encoder.[3] A quick search over length-7 permutations yields many permutations that transform all weight-6 and weight-7 codewords from the outer encoder into non-terminating input sequences to the inner encoder. We chose to use the permutation

$$P_7 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{28}$$

for this example. Table 4 lists the results of this permutation.

4.5.2 Dynamical System Representation with Sub-vector Interleaving

We can incorporate the sub-vector interleaver, P_7, in (28) into the dynamical system representation of the entire SCCC. To do this, it is simplest to consider encoding entire length-7 sub-vectors of the input sequence, $\underline{m}$, at each iteration of the dynamical

[3] If there were no weight-6 or weight-7 terminating input sequences to the inner encoder corresponding to terminated codewords from the outer encoder, then the sub-vector interleaver could simply be an identity permutation.

system. The dynamical system representation for the outer encoder in this case is given by

$$\begin{bmatrix} v_{o1}(n+7) \\ v_{o2}(n+7) \\ v_{o3}(n+7) \end{bmatrix} = \begin{bmatrix} 1\ 0\ 0 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \end{bmatrix} \begin{bmatrix} v_{o1}(n) \\ v_{o2}(n) \\ v_{o3}(n) \end{bmatrix} + [V_{3,7}] \begin{bmatrix} m(n) \\ m(n+1) \\ m(n+2) \\ m(n+3) \\ m(n+4) \\ m(n+5) \\ m(n+6) \end{bmatrix},$$

where $\underline{v}_o = \begin{bmatrix} v_{o1}(n)\ v_{o2}(n)\ v_{o3}(n) \end{bmatrix}$ is the state of the outer encoder at time n, and

$$V_{3,7} = \begin{bmatrix} 1\ 0\ 0\ 1\ 1\ 1\ 0 \\ 0\ 0\ 1\ 1\ 1\ 0\ 1 \\ 0\ 1\ 1\ 1\ 0\ 1\ 0 \end{bmatrix}$$

is a matrix containing one period of each of the delay state impulse responses of the outer or inner encoder in each of its rows.

The inner encoder encodes a concatenation of the input sequence to the outer encoder, $\underline{m}$, and the output sequence generated by the outer encoder, $\underline{y}$. We can express a length-7 sub-vector of the output sequence of the outer encoder,

$$\underline{y} = \begin{bmatrix} y(n)\ y(n+1)\ y(n+2)\ y(n+3)\ y(n+4)\ y(n+5)\ y(n+6) \end{bmatrix},$$

as a function of a length-7 sub-vector of the input sequence, $\underline{m}$, and the state of the outer encoder at time n:

$$\begin{bmatrix} y(n) \\ y(n+1) \\ y(n+2) \\ y(n+3) \\ y(n+4) \\ y(n+5) \\ y(n+6) \end{bmatrix} = \begin{bmatrix} 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0 \\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0 \\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0 \\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 0 \\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 0 \\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0 \\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 1 \end{bmatrix} \begin{bmatrix} v_{o1}(n) \\ v_{o2}(n) \\ v_{o3}(n) \\ m(n) \\ m(n+1) \\ m(n+2) \\ m(n+3) \\ m(n+4) \\ m(n+5) \\ m(n+6) \end{bmatrix}. \tag{29}$$

If we consider encoding a length-7 sub-vector of the input sequence, $\underline{m}$, and a length-7 sub-vector of the output sequence from the outer encoder, $\underline{y}$, at each iteration

of the dynamical system representation of the inner encoder, then the state of the inner encoder can be expressed as

$$\begin{bmatrix} v_{i1}(n+7) \\ v_{i2}(n+7) \\ v_{i3}(n+7) \end{bmatrix} = \begin{bmatrix} v_{i1}(n) \\ v_{i2}(n) \\ v_{i3}(n) \end{bmatrix} + \begin{bmatrix} V_{3,7} & V_{3,7} \end{bmatrix} \cdot \begin{bmatrix} P_7 & 0 \\ 0 & P_7 \end{bmatrix} \cdot \underbrace{\begin{bmatrix} m(n) \\ y(n) \\ m(n+1) \\ y(n+1) \\ m(n+2) \\ y(n+2) \\ m(n+3) \\ y(n+3) \\ m(n+4) \\ y(n+4) \\ m(n+5) \\ y(n+5) \\ m(n+6) \\ y(n+6) \end{bmatrix}}_{\underline{p}} . \tag{30}$$

The sub-vector interleaver, P_7, we have chosen permutes the input sequence to the inner encoder, $\underline{p}$, as follows:

$$\begin{bmatrix} P_7 & 0 \\ 0 & P_7 \end{bmatrix} \cdot \underbrace{\begin{bmatrix} m(n) \\ y(n) \\ m(n+1) \\ y(n+1) \\ m(n+2) \\ y(n+2) \\ m(n+3) \\ y(n+3) \\ m(n+4) \\ y(n+4) \\ m(n+5) \\ y(n+5) \\ m(n+6) \\ y(n+6) \end{bmatrix}}_{\underline{p}} = \underbrace{\begin{bmatrix} y(n) \\ m(n) \\ y(n+2) \\ y(n+1) \\ m(n+3) \\ m(n+1) \\ m(n+2) \\ m(n+4) \\ y(n+3) \\ m(n+6) \\ m(n+5) \\ y(n+6) \\ y(n+4) \\ y(n+5) \end{bmatrix}}_{\underline{p}_\pi} . \tag{31}$$

Combining equations (29), (30), and (31) gives us the following expression for the state of the inner encoder at time $(n+7)$ as a function of the state of the inner encoder at time n and a length-7 sub-vector of the input sequence to the outer encoder, $\begin{bmatrix} m(n) & m(n+1) & m(n+2) & m(n+3) & m(n+4) & m(n+5) & m(n+6) \end{bmatrix}$:

$$\begin{bmatrix} v_{i1}(n+7) \\ v_{i2}(n+7) \\ v_{i3}(n+7) \\ v_{o1}(n+7) \\ v_{o2}(n+7) \\ v_{o3}(n+7) \end{bmatrix} = \begin{bmatrix} 1&0&0&1&1&0 \\ 0&1&0&1&1&1 \\ 0&0&1&1&1&1 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix} \begin{bmatrix} v_{i1}(n) \\ v_{i2}(n) \\ v_{i3}(n) \\ v_{o1}(n) \\ v_{o2}(n) \\ v_{o3}(n) \end{bmatrix}$$
$$+ \begin{bmatrix} 0&0&0&0&1&1&0 \\ 0&0&0&0&1&1&1 \\ 1&0&1&1&1&1&0 \\ 0&0&1&1&1&0&1 \\ 0&1&1&1&0&1&0 \\ 1&1&1&0&1&0&0 \end{bmatrix} \begin{bmatrix} m(n) \\ m(n+1) \\ m(n+2) \\ m(n+3) \\ m(n+4) \\ m(n+5) \\ m(n+6) \end{bmatrix}.$$

Including the input sequence in the states of the dynamical system representation of the SCCC yields the following autonomous dynamical system representation of the SCCC:

$$\begin{bmatrix} \underline{v}_i(n+7) \\ \underline{v}_o(n+7) \\ \underline{m}(n+7) \end{bmatrix} = \underbrace{\begin{bmatrix} 1&0&0&1&1&0&0&0&0&0&1&1&0&0&\cdots&0 \\ 0&1&0&1&1&1&0&0&0&0&1&1&1&0&\cdots&0 \\ 0&0&1&1&1&1&1&0&1&1&1&1&1&0&\cdots&0 \\ 0&0&0&1&0&0&0&0&1&1&1&0&1&0&\cdots&0 \\ 0&0&0&0&1&0&0&1&1&1&0&1&0&0&\cdots&0 \\ 0&0&0&0&0&1&1&1&1&0&1&0&0&0&\cdots&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&&0 \\ & & & & & & & & & & & & & & \ddots & \\ & & & & & & & & & & & & & & & 1 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&\cdots&0 \\ \vdots & & & & & & & & & & & & & & & \vdots \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&\cdots&0 \end{bmatrix}}_{B} \begin{bmatrix} \underline{v}_i(n) \\ \underline{v}_o(n) \\ \underline{m}(n) \end{bmatrix},$$

where

$$\underline{v}_i'(n) = \begin{bmatrix} v_{i1}(n) & v_{i2}(n) & v_{i3}(n) \end{bmatrix},$$

$$\underline{v}_o'(n) = \begin{bmatrix} v_{o1}(n) & v_{o2}(n) & v_{o3}(n) \end{bmatrix},$$

$\underline{m}(n)$ contains the entire length-N_{out} input sequence at $n = 0$:

$$\underline{m}'(n) = \begin{bmatrix} m(1) & m(2) & \cdots & m(N_{out}-1) & m(N_{out}) \end{bmatrix},$$

and $\underline{m}\,(N_{out})$ is a length-N_{out} zero vector:

$$\underline{m}'\,(N_{out}) = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

Iterating the autonomous dynamical system representation $\frac{N_{out}}{7}$ times, which corresponds to the time after which the last length-7 sub-vector of the input sequence $\underline{m}$ has been encoded, allows the state of the outer and inner encoder at time N_{out} to be expressed as a mapping of the input sequence $\underline{m}$ by a matrix containing the delay state impulse responses of the outer and inner encoder in each of its rows:

$$\begin{bmatrix} \underline{v}_i\,(N_{out}) \\ \underline{v}_o\,(N_{out}) \end{bmatrix} = \begin{bmatrix} V_{6,14} & \cdots & V_{6,14} \end{bmatrix} \cdot \underline{m}. \tag{32}$$

The matrix $V_{6,14}$ contains one period of each of the delay state impulse responses of the inner encoder in each of the first three rows (the period of the inner encoder's delay state outputs in response to an impulse to the *outer* encoder is 14), and two periods of the outer encoder's delay state impulse responses in the next three rows:

$$V_{6,14} = \begin{bmatrix} 0&0&0&0&1&1&0&0&1&0&0&0&0&1 \\ 0&0&0&0&1&1&1&1&0&1&0&1&0&0 \\ 1&0&1&1&1&1&0&0&0&0&1&1&0&1 \\ 0&0&1&1&1&0&1&0&0&1&1&1&0&1 \\ 0&1&1&1&0&1&0&0&1&1&1&0&1&0 \\ 1&1&1&0&1&0&0&1&1&1&0&1&0&0 \end{bmatrix}. \tag{33}$$

4.5.3 Minimal Length Terminating Tail Sequence

From (32), we see that a terminating input sequence $\underline{m}_t$, which leaves the outer and inner encoder in the zero state after its last non-zero bit has been encoded, is mod 2 orthogonal to each of the delay states of the outer and inner encoders:

$$\begin{bmatrix} V_{6,14} & \cdots & V_{6,14} \end{bmatrix} \cdot \underline{m}_t = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The mod 2 sum of length-14 sub-vectors of an input sequence $\underline{m}$ is given by

$$\underline{m}'_{\Sigma} = \begin{bmatrix} m_{\Sigma_1} & m_{\Sigma_2} & \cdots & m_{\Sigma_{13}} & m_{\Sigma_{14}} \end{bmatrix},$$

and the mod 2 sum of length-14 sub-vectors of a terminating input sequence $\underline{m_t}$ is given by

$$\underline{m'_{\Sigma_t}} = \left[m_{\Sigma_{t1}} \; m_{\Sigma_{t2}} \; \cdots \; m_{\Sigma_{t13}} \; m_{\Sigma_{t14}} \right].$$

From (32) and (33), we see that $\underline{m_{\Sigma_t}}$ that satisfies the following:

$$\begin{bmatrix} m_{\Sigma_{t5}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t9}} \oplus m_{\Sigma_{t14}} \\ m_{\Sigma_{t5}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t7}} \oplus m_{\Sigma_{t8}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t12}} \\ m_{\Sigma_{t1}} \oplus m_{\Sigma_{t3}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t12}} \oplus m_{\Sigma_{t14}} \\ m_{\Sigma_{t3}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t7}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t12}} \oplus m_{\Sigma_{t14}} \\ m_{\Sigma_{t2}} \oplus m_{\Sigma_{t3}} \oplus m_{\Sigma_{t4}} \oplus m_{\Sigma_{t6}} \oplus m_{\Sigma_{t9}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t11}} \oplus m_{\Sigma_{t13}} \\ m_{\Sigma_{t1}} \oplus m_{\Sigma_{t2}} \oplus m_{\Sigma_{t3}} \oplus m_{\Sigma_{t5}} \oplus m_{\Sigma_{t8}} \oplus m_{\Sigma_{t9}} \oplus m_{\Sigma_{t10}} \oplus m_{\Sigma_{t12}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{34}$$

Therefore, a length-6 tail can be appended to the end of a length-$(N_{out} - 6)$ input sequence, $\underline{m}$, that will convert $\underline{m}$ to a terminating input sequence, $\underline{m_t}$:

$$\underline{m'_t} = \left[m(1) \; m(2) \; \cdots \underbrace{m(N_{out} - 5) \; m(N_{out} - 4) \; \cdots \; m(N_{out} - 1) \; m(N_{out})}_{\text{terminating tail sequence, } \underline{m_{tail}}} \right].$$

The relationship between $\underline{m_\Sigma}$ and $\underline{m_{\Sigma_t}}$ in this case is expressed as:

$$\begin{bmatrix} m_{\Sigma_1} \\ m_{\Sigma_2} \\ \vdots \\ m_{\Sigma_9} + m(N_{out} - 5) \\ m_{\Sigma_{10}} + m(N_{out} - 4) \\ m_{\Sigma_{11}} + m(N_{out} - 3) \\ m_{\Sigma_{12}} + m(N_{out} - 2) \\ m_{\Sigma_{13}} + m(N_{out} - 1) \\ m_{\Sigma_{14}} + m(N_{out}) \end{bmatrix} = \begin{bmatrix} m_{\Sigma_{t1}} \\ m_{\Sigma_{t2}} \\ \vdots \\ m_{\Sigma_{t9}} \\ m_{\Sigma_{t10}} \\ m_{\Sigma_{t11}} \\ m_{\Sigma_{t12}} \\ m_{\Sigma_{t13}} \\ m_{\Sigma_{t14}} \end{bmatrix}. \tag{35}$$

Therefore, from (34) and (35) we see that a length-6 terminating tail sequence must satisfy:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} m(N_{out} - 5) \\ m(N_{out} - 4) \\ m(N_{out} - 3) \\ m(N_{out} - 2) \\ m(N_{out} - 1) \\ m(N_{out}) \end{bmatrix} \tag{36}$$

$$= \begin{bmatrix} m_{\Sigma_5} \oplus m_{\Sigma_6} \oplus m_{\Sigma_9} \oplus m_{\Sigma_{14}} \\ m_{\Sigma_5} \oplus m_{\Sigma_6} \oplus m_{\Sigma_7} \oplus m_{\Sigma_8} \oplus m_{\Sigma_{10}} \oplus m_{\Sigma_{12}} \\ m_{\Sigma_1} \oplus m_{\Sigma_3} \oplus m_{\Sigma_4} \oplus m_{\Sigma_5} \oplus m_{\Sigma_6} \oplus m_{\Sigma_{11}} \oplus m_{\Sigma_{12}} \oplus m_{\Sigma_{14}} \\ m_{\Sigma_3} \oplus m_{\Sigma_4} \oplus m_{\Sigma_5} \oplus m_{\Sigma_7} \oplus m_{\Sigma_{10}} \oplus m_{\Sigma_{11}} \oplus m_{\Sigma_{12}} \oplus m_{\Sigma_{14}} \\ m_{\Sigma_2} \oplus m_{\Sigma_3} \oplus m_{\Sigma_4} \oplus m_{\Sigma_6} \oplus m_{\Sigma_9} \oplus m_{\Sigma_{10}} \oplus m_{\Sigma_{11}} \oplus m_{\Sigma_{13}} \\ m_{\Sigma_1} \oplus m_{\Sigma_2} \oplus m_{\Sigma_3} \oplus m_{\Sigma_5} \oplus m_{\Sigma_8} \oplus m_{\Sigma_9} \oplus m_{\Sigma_{10}} \oplus m_{\Sigma_{12}} \end{bmatrix}, \tag{37}$$

which can be rewritten to show that

$$\begin{bmatrix} m(N_{out}-5) \\ m(N_{out}-4) \\ m(N_{out}-3) \\ m(N_{out}-2) \\ m(N_{out}-1) \\ m(N_{out}) \end{bmatrix} = \begin{bmatrix} m_{\Sigma_1} \oplus m_{\Sigma_2} \oplus m_{\Sigma_3} \oplus m_{\Sigma_6} \oplus m_{\Sigma_7} \oplus m_{\Sigma_9} \\ m_{\Sigma_1} \oplus m_{\Sigma_6} \oplus m_{\Sigma_7} \oplus m_{\Sigma_{10}} \\ m_{\Sigma_1} \oplus m_{\Sigma_2} \oplus m_{\Sigma_4} \oplus m_{\Sigma_5} \oplus m_{\Sigma_6} \oplus m_{\Sigma_7} \oplus m_{\Sigma_8} \oplus m_{\Sigma_{11}} \\ m_{\Sigma_1} \oplus m_{\Sigma_5} \oplus m_{\Sigma_8} \oplus m_{\Sigma_{12}} \\ m_{\Sigma_1} \oplus m_{\Sigma_2} \oplus m_{\Sigma_5} \oplus m_{\Sigma_7} \oplus m_{\Sigma_8} \oplus m_{\Sigma_{13}} \\ m_{\Sigma_1} \oplus m_{\Sigma_2} \oplus m_{\Sigma_3} \oplus m_{\Sigma_5} \oplus m_{\Sigma_7} \oplus m_{\Sigma_{14}} \end{bmatrix}.$$

4.5.4 Weight Distribution Analysis

Using the method outlined in [22], we were able to compute free distance codeword weight an SCCC employing a particular length-N_{in} interleaver. We performed this analysis for the SCCC in Fig. 16 for a large number ($\approx$ 100000) of interleavers to find the average free distance codeword weight for this SCCC. The result of this analysis is shown in Fig. 17 which compares the average free distance codeword weight of the SCCC employing either a random or a sub-vector interleaver as a function of interleaver length. These results show that sub-vector interleaving increases the average free distance codeword weight.

Extending the method outlined in [22], we were able to compute expected multiplicity for codewords of the SCCC in Fig. 12 as a function of the codeword's Hamming weight. The results of this analysis are presented in Table 5 which lists the expected input sequence weight, $E[w]$, and expected multiplicity, $E[mult]$, of codewords with Hamming weight ≤ 20 assuming a length-231 average random and average sub-vector interleaver. These results allowed us to compute the BER contribution of error patterns corresponding to weight-$(d+p)$ codewords generated by weight-w input sequences:

$$P_b\left(e \mid \underline{e}_{w,d,p}\right) \approx \left(\tfrac{E[w]}{N_{out}}\right) \cdot E[mult] \cdot Q\left(\sqrt{(d+p) \cdot \tfrac{2R_c\varepsilon_b}{N_0}}\right).$$

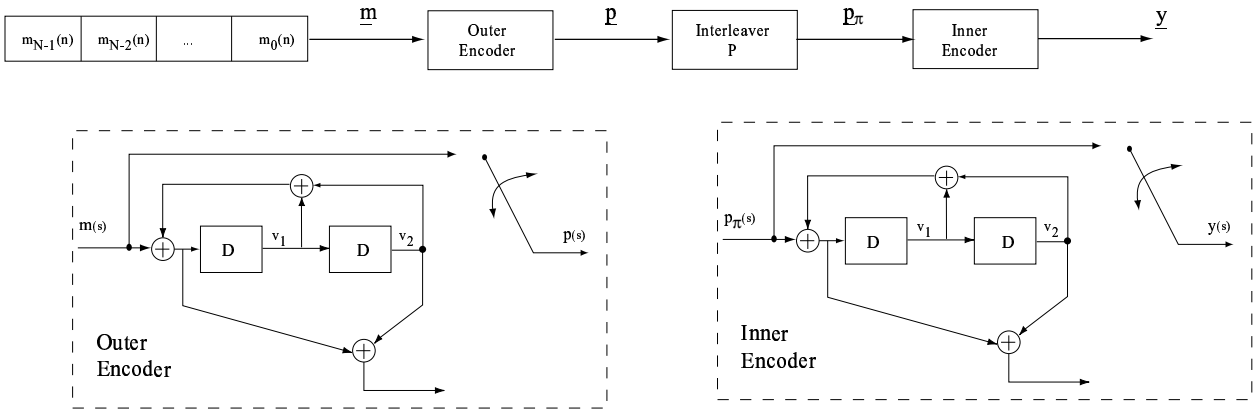

Figure 16: A diagram of an SCCC employing identical rate-$\frac{1}{2}$ recursive convolutional encoders with two delay states.

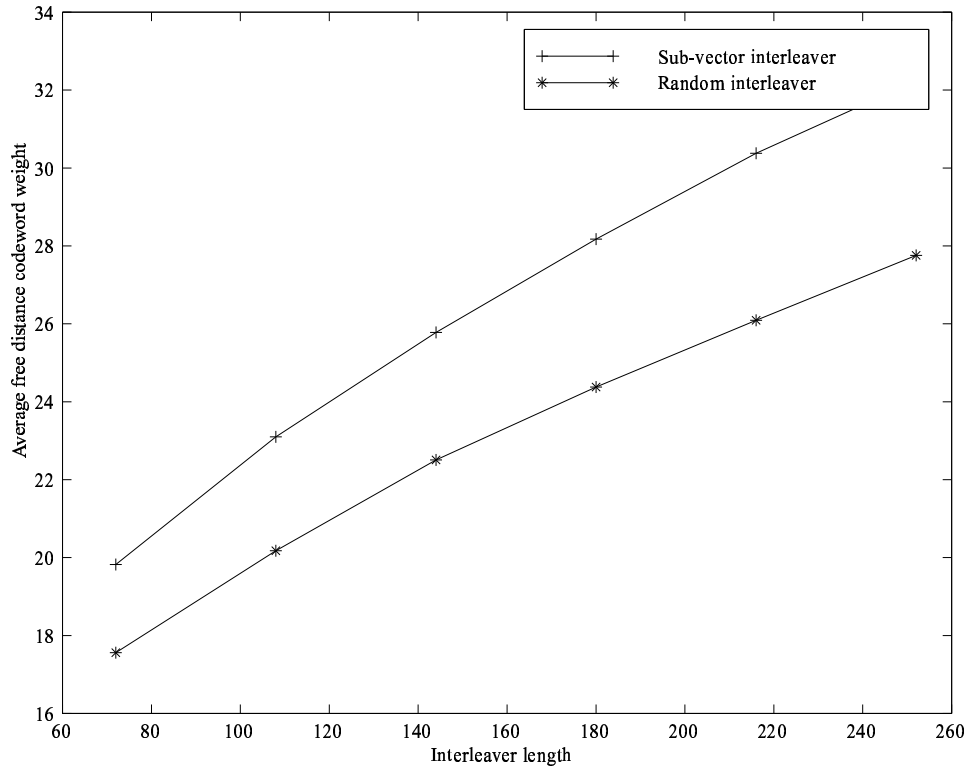

Figure 17: Average free distance codeword weight of the SCCC in Fig. 16 employing a sub-vector interleaver and an average random interleaver as a function of interleaver length.

Table 5: Weight distribution up to weight 20 of SCCC in Fig. 12 employing average length-231 random and sub-vector interleavers. The codeword weight, d, expected input sequence weight, $E[w]$, and expected multiplicity, $E[mult]$, was computed from an ensemble of $100,000$ randomly generated interleavers.

	Random		*Sub-vector*	
d	$E[w]$	$E[mult]$	$E[w]$	$E[mult]$
6	3.56	$\frac{9}{10000}$	–	-
7	1.98	$\frac{506}{10000}$	–	-
8	3.64	$\frac{33}{10000}$	–	-
9	2.06	$\frac{369}{10000}$	4.09	$\frac{5.5}{10000}$
10	3.49	$\frac{129}{10000}$	4.44	$\frac{4.5}{10000}$
11	2.08	$\frac{1013}{10000}$	3.97	$\frac{18.4}{10000}$
12	3.38	$\frac{325}{10000}$	4.00	$\frac{10}{10000}$
13	1.86	$\frac{925}{10000}$	4.40	$\frac{100}{10000}$
14	4.04	$\frac{575}{10000}$	4.83	$\frac{120}{10000}$
15	2.47	$\frac{2950}{10000}$	4.35	$\frac{460}{10000}$
16	3.11	$\frac{2325}{10000}$	5.36	$\frac{250}{10000}$
17	3.02	$\frac{2150}{10000}$	5.20	$\frac{710}{10000}$
18	4.34	$\frac{2900}{10000}$	4.95	$\frac{1280}{10000}$
19	3.42	$\frac{5600}{10000}$	5.01	$\frac{2070}{10000}$
20	3.83	$\frac{6725}{10000}$	5.58	$\frac{1610}{10000}$

In Fig. 18 we plot the BER contribution of error patterns corresponding to codewords with Hamming weight $d + p \leq 11$. Our weight distribution analysis shows us that the BER contribution of the codewords with Hamming weight $d + p \leq 8$ can be eliminated by sub-vector interleaving, and the multiplicity of other low Hamming weight codewords is reduced. Results of a Monte Carlo simulation of this SCCC which confirms this analysis will be shown in Sec. 4.6.

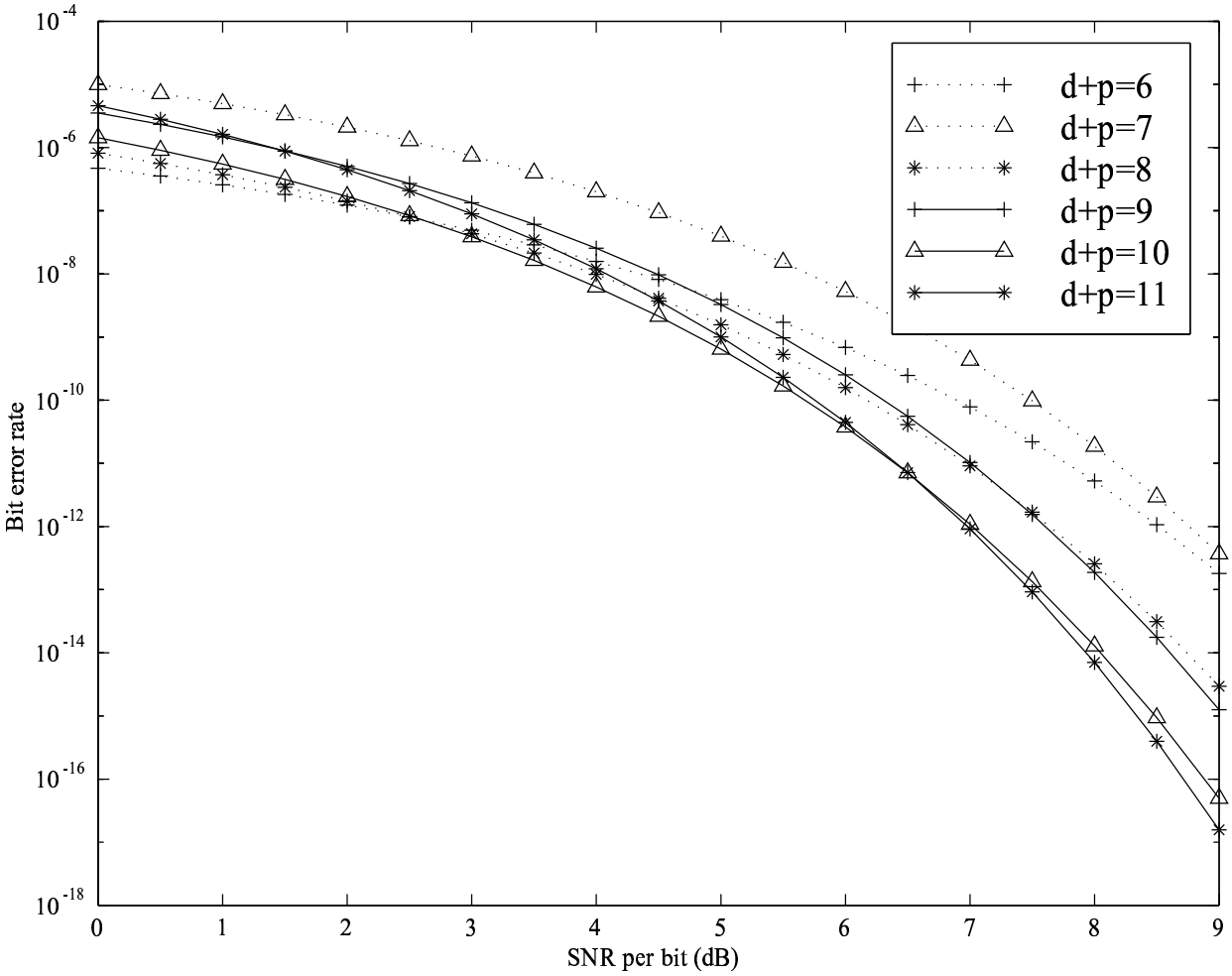

Figure 18: BER contribution of error patterns corresponding to codewords with Hamming weight $d + p \leq 11$ for the SCCC in Fig. 12 and a length-231 random interleaver. Sub-vector interleaving eliminates the BER contribution of the error patterns corresponding to codewords with Hamming weight $d + p \leq 8$.

4.6 Simulation Results of a Serially Concatenated Turbo Coding Scheme

Because of the difficulty of performing Monte Carlo simulations of coding schemes with extremely low BERs, we have chosen to design an interleaver for a relatively weak SCCC with a length-154 input sequence (including the tail). As shown in Fig. 12, we employ a rate $\frac{2}{3}$ non-systematic convolutional encoder with memory = 2 as the outer encoder and a rate $\frac{1}{2}$ systematic recursive convolutional encoder as the inner encoder in our simulation. Since we have a rate $\frac{2}{3}$ outer encoder, a length-154 input sequence requires a length-231 interleaver.

The minimum weight of a codeword from the outer encoder in this SCCC is $d_{free} = 3$. However, the free distance codeword in this scheme is generated by a weight-4 codeword from the outer encoder that is interleaved into the input sequence to the inner encoder

$$\underline{p}_{\pi} = \left[1\ 1\ 1\ 0\ 1 \right]$$

which generates the parity sequence output

$$\underline{y} = \left[1\ 0\ 0\ 0\ 1 \right].$$

The overall weight of this codeword is then $d + p = 4 + 2 = 6$. By sub-vector interleaving, we were able to cause all weight-3 and weight-4 input sequences to the inner encoder to correspond to codewords with non-terminating parity sequences from the inner encoder. Table 6 list the weight-5 and weight-6 terminating input sequences to the inner encoder for our sub-vector interleaver.

In our simulation with a length-231 interleaver, the most significant error pattern for the sub-vector interleaver, in terms of its contribution to the BER at low SNRs, of the SCCC employing our sub-vector interleaver is a single terminating error pattern. It corresponds to a codeword with a weight-5 output sequence from the outer encoder

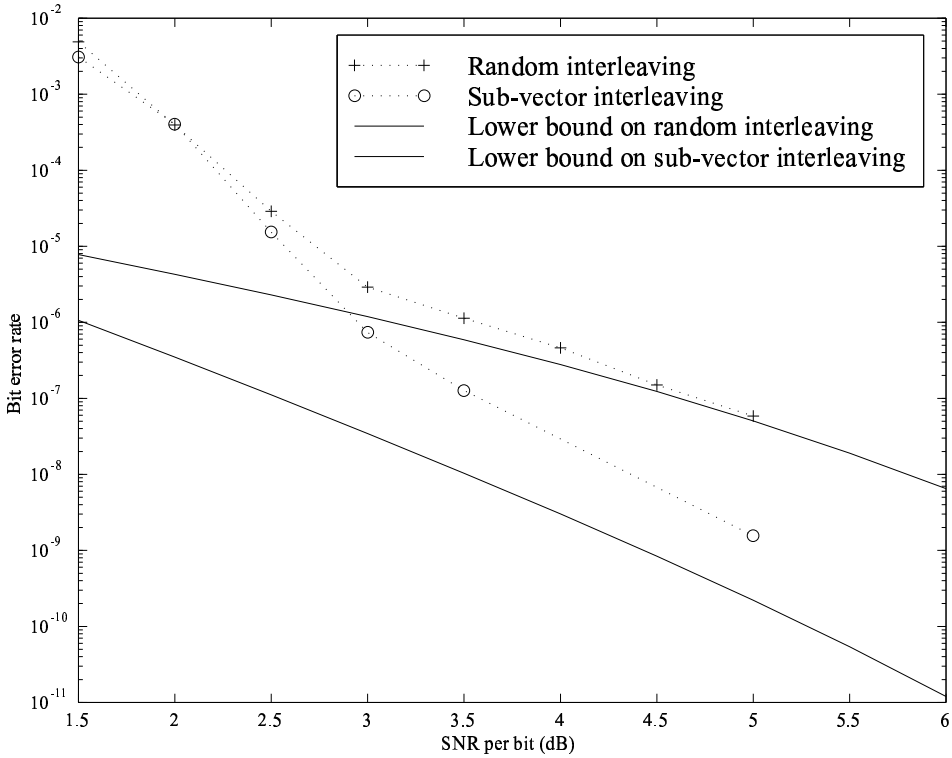

Figure 19: Simulated BER plots for the SCCC shown in Fig. 12 and a random interleaver and the same scheme employing a sub-vector interleaver. The input sequence length is 154 in both cases, resulting in a length-231 interleaver. The simulated BERs are plotted along with their lower bounds.

and a weight-4 output sequence from the inner encoder. Such a codeword has total weight $d + p = 5 + 4 = 9$. Table 7 shows the outer encoder output sequence $\underline{p}$ interleaved into the inner encoder input sequence $\underline{p}_\pi$ that generates the free distance codeword for the sub-vector interleaved SCCC.

Our simulation shows that we were able to decrease the BER of the sub-vector interleaved SCCC significantly over the same scheme employing random interleaving. These results are shown in Fig. 19. We see that the BER in the randomly interleaved simulation converges to its lower bound at 4.5 dB. The BER of the sub-vector interleaved simulation shows approximately 1.75 dB improvement over the average random interleaver at an SNR of 3 dB and has begun to converge to its lower bound at an SNR of 5 dB.

Table 6: A table of outer encoder terminating input sequences and terminated output sequences that correspond to weight-5 and weight-6 terminating input sequences to the inner encoder for the SCCC in Fig. 12.

Input Sequence, $\underline{m}$		Output Sequence, $\underline{p}$
$\begin{bmatrix}1&0&1&0&0&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}0&1&0&1&0&1&0\\1&0&0&1&0&0&0\end{bmatrix}$
$\begin{bmatrix}1&1&0&1&1&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}1&1&0&0&0&0&0\\1&0&0&1&1&0&0\end{bmatrix}$
$\begin{bmatrix}0&1&1&0&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}0&1&1&0&0&1&1\\1&1&0&0&0&0&0\end{bmatrix}$
$\begin{bmatrix}1&1&1&0&1&1&0&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}1&1&0&1&1&0&0\\0&1&0&0&0&1&0\end{bmatrix}$
$\begin{bmatrix}1&0&1&1&0&1&1&0&0&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}0&1&0&1&0&0&1\\0&0&0&0&0&1&1\\0&0&1&0&0&0&0\end{bmatrix}$
$\begin{bmatrix}0&1&1&1&0&1&1&0&1&1&0&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}0&1&1&0&1&0&0\\0&0&0&0&1&0&0\\1&0&0&0&1&0&0\end{bmatrix}$
$\begin{bmatrix}1&0&0&1&1&1&0&1&1&1&0&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}1&0&1&1&0&0&0\\1&0&0&0&0&0&1\\0&0&0&0&1&0&0\end{bmatrix}$

Table 7: The free distance codeword for the SCCC in Fig. 12 employing a sub-vector interleaver has Hamming weight $= d + p = 5 + 4 = 9$.

Outer Encoder Output Sequence, $\underline{p}$		Inner Encoder Permuted Input Sequence, $\underline{p}_\pi$		Inner Encoder Output Sequence, $\underline{y}$
$\begin{bmatrix}1&1&0&0&0&0&0\\1&0&0&1&1&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}1&1&0&1&1&0&0\\1&0&0&0&0&0&0\end{bmatrix}$	$\rightarrow$	$\begin{bmatrix}1&0&1&0&0&1&0\\1&0&0&0&0&0&0\end{bmatrix}$

4.7 Summary

Sub-vector interleavers significantly improve the performance of SCCCs compared to SCCCs employing an average random interleaver. By terminating both the inner and outer encoders in the zero state and increasing the minimum weight of codewords from the outer encoder that are interleaved into codewords with terminating parity sequences in the inner encoder, sub-vector interleavers both:

- Decrease the BER floor of the SCCC by increasing the average free distance codeword weight.
- Increase the interleaver gain of the SCCC.

The key feature of the method of interleaving by sub-vector is its low-complexity design, which typically involves searching over interleavers equal in length to only a few periods of the feedback polynomial of the inner recursive convolutional encoder. Sub-vector interleaver design results in a type of block interleaver in which rows are permuted in order to eliminate the error patterns that are most significant in terms of their contribution to the BER. By incorporating random row interleaving into the design of sub-vector interleavers, we increase the gain in performance expected with increased interleaver length and are able to scale up our design to form any length-N_{in} interleaver without additional computation.

The contributions of the research described in this chapter are as follows:

- Dynamical system representation of SCCCs.
- Sub-vector interleaving for SCCCs, which extends to the benefits of sub-vector interleaving for PCCCs to SCCCs:
  - low complexity design,
  - no additional complexity to the encoding or decoding,

– lowering of the error floor of the SCCC,

– ability to be scaled up to any length-N interleaver without additional design complexity,

– Self-termination of the inner and outer encoders in an SCCC, allowing for increased interleaver gain when designing sub-vector interleavers for SCCCs.

– increase in the average free distance codeword weight for the coding scheme employing a sub-vector interleaver with randomly generated row permutations.

CHAPTER V

SUB-VECTOR CONSTRAINED S-RANDOM INTERLEAVING

The S-random algorithm was developed in [17] to generate permutations of any length, long or short, that perform well for use with either PCCCs or SCCCs. The S-random algorithm seeks to permute elements in the input sequence that are separated by a distance less than S to positions in the output sequence that are separated by a distance greater than S. The idea behind the S-random algorithm is that short terminating input sequences of length $< S$ will be "broken" by the S-random interleaver and will become either long terminating or long non-terminating input sequences which will in turn generate high weight parity sequence outputs for the coding scheme.

We would like to use the notation

$$\underline{\pi}_N = \left[\pi_0 \ \pi_1 \ \cdots \ \pi_{N-2} \ \pi_{N-1} \right]$$

to describe an interleaver that operates on a length-N sequence where the i^{th} element of the interleaved sequence is the π_i^{th} element of the input sequence.

Using this notation, the S-random algorithm is described as follows:

1. Randomly select the first element of the permutation, π_0, such that

$$\pi_0 \in \{0, 1, \ldots N-1\}.$$

2. For each subsequent element of the permutation, π_n, randomly select another integer from the set

$$\{0, 1, \ldots N-1\} \setminus \{\pi_0, \pi_1, \ldots \pi_{n-1}\}$$

and compare it to the S previously selected elements $\{\pi_{n-S}, \pi_{n-S+1}, \ldots \pi_{n-1}\}$.

3. If $|\pi_n - \pi_{n-m}| \geq S$ for all $m \in \{1, 2, \ldots, S\}$, set $\underline{\pi}_N(n) = \pi_n$. Otherwise, repeat the previous step.

4. Continue this iterative process until the permutation is complete with every element satisfying the S-random criteria.

Though the S-random algorithm is simple to implement and computationally efficient, it is a generic algorithm in the sense that it does not consider the constituent encoders of the PCCC or SCCC for which it is designed. This drawback can be addressed by incorporating S-random interleaving into the sub-vector interleaver design algorithm.

Incorporating concepts from S-random interleaving into the sub-vector interleaving method leads to an improvement on both of these methods of interleaver design (see Fig. 20 for an illustration) previously described in [34]. This variant of sub-vector interleaving employs S-random, instead of random, interleavers over the columns of the sub-vector interleaver to design a length-N interleaver. The algorithm for computing the S-random column interleavers can be described as a sub-vector constrained version of the original S-random algorithm:

1. Permute the columns of a matrix with the integers $\{0, 1, \ldots, N-1\}$ read in row-wise using the length-η interleaver

$$\underline{\pi}_\eta = \left[\pi_0 \ \pi_1 \ \cdots \ \pi_{\eta-2} \ \pi_{\eta-1} \right].$$

2. Randomly select the first element of the permutation, $\pi_{0,0}$, such that[1]

$$\pi_{0,0} \in \{\pi_0, \pi_0 + \eta, \pi_0 + 2\eta, \ldots, \pi_0 + N - \eta\}.$$

[1]We use the notation $\pi_{m,n}$ to designate the interleaver element in the m^{th} row of the n^{th} column of the sub-vector interleaver, $\underline{\pi}_N(m, n)$.

3. For each subsequent element of the permutation, $\pi_{m,n}$, randomly select another integer from the set

$$\{\pi_n, \pi_n + \eta, \pi_n + 2\eta, \ldots, \pi_n + N - \eta\} \setminus \{\pi_{0,n}, \pi_{1,n}, \ldots, \pi_{m-1,n}\}$$

and compare it to the S previously selected elements.

4. If distance between $\pi_{m,n}$ and the S previously selected elements is $\geq S$, set $\underline{\pi}_N(m,n) = \pi_{m,n}$. Otherwise, repeat the previous step.

5. Proceed row-wise through the sub-vector interleaved sequence until the permutation is complete with every element satisfying the sub-vector constrained S-random criteria.

Note that this method of designing constrained S-random interleaving considers the relative distance between interleaved elements in different columns of the sub-vector interleaved sequence when computing the S-random row interleavers. Incorporating sub-vector interleaving does not increase the complexity of the S-random algorithm, though it may cause the algorithm to converge more slowly to the final solution than the straight S-random algorithm.

5.1 S Parameter

The value of S chosen for an S-random interleaver is an important design consideration. The larger the value of S chosen, the greater the spreading of the input sequence. However, increased values of S lead to increased time to convergence of the S-random algorithm. Designers typically choose a value of S less than the square root of half the interleaver length, $S < \sqrt{\frac{N}{2}}$. This value was suggested in [17] where the S-random algorithm was originally described since it leads to good spreading in the resulting interleaver and reasonable computational time for the S-random algorithm.

Incorporating S-random interleaving into sub-vector interleaving puts an additional constraint on the S-random algorithm since the pseudo-random search for

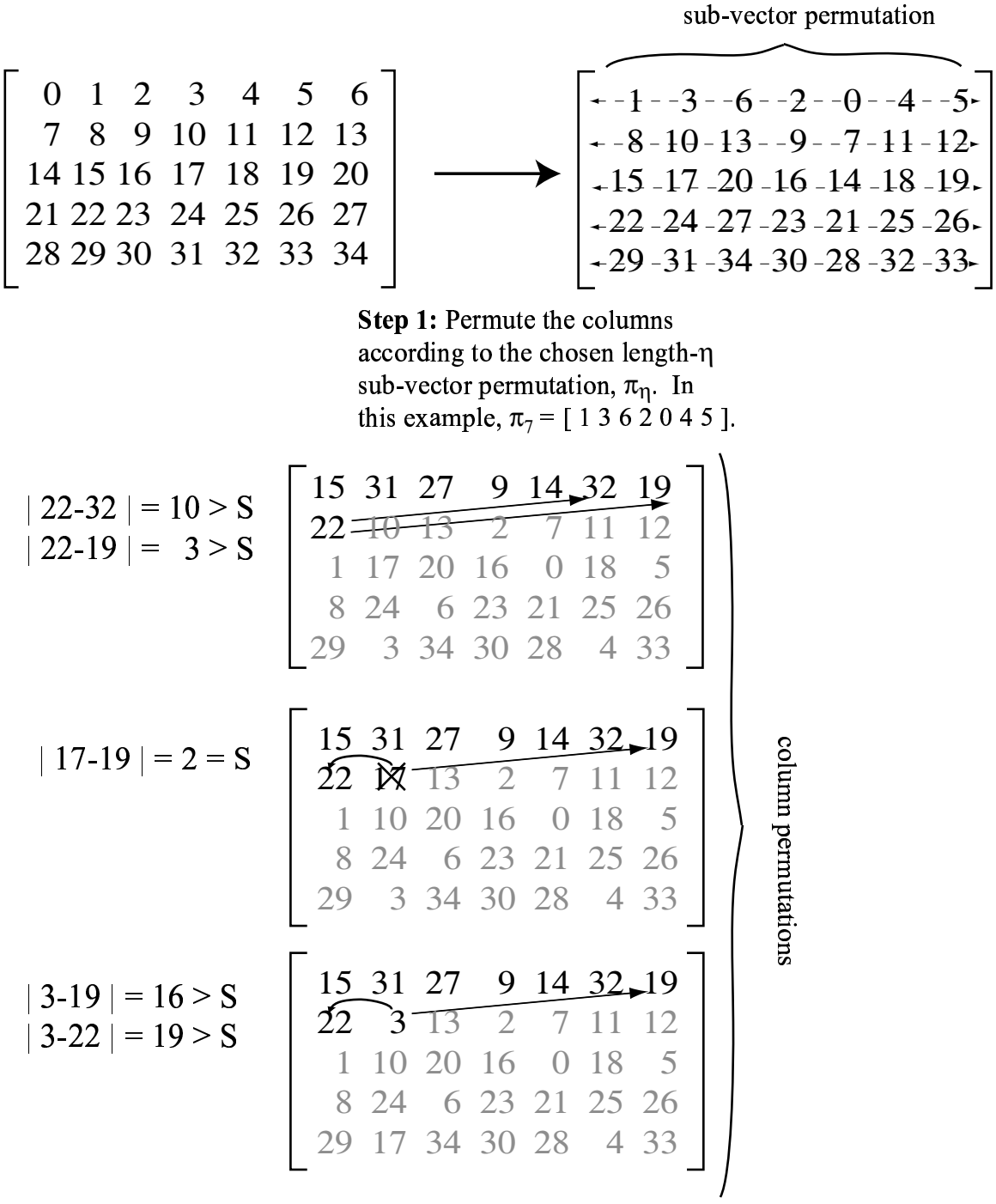

Figure 20: Sub-vector constrained S-random algorithm, $S = 2$. Sub-vector length $= 7$.

subsequent values of the interleaver are limited to a fraction $\frac{N_r}{\eta}$ of the remaining possible choices, where N_r is the remaining number of interleaver values and η is the length of the sub-vector.

In Section 5.1.1, we will discuss the maximum possible value of $S = S_{\max}$ that can be achieved in an S-random interleaver. We will describe a systematic method for obtaining an $S_{\max}$-random interleaver (note that some similar ideas for generating interleavers to protect convolutional coding schemes from burst errors were described by Dunscome, et al., in [18]). In Section 5.1.2 we will discuss the impact of constraining the S-random algorithm to a sub-vector interleaver on the value of $S_{\max}$.

5.1.1 Increasing S for S-random Interleavers

We can modify the S-random algorithm slightly by allowing two parameters to describe the algorithm: S_a and S_b. The modified version of the S-random algorithm is as follows:

1. Select a randomly generated integer as the first element of the permutation.

2. For each subsequent element of the permutation, randomly generate another integer and compare it to the S_a previously selected elements.

3. If the current selected element of the permutation is within a distance S_b of any of the S_a previously selected elements, discard it and randomly select another integer.

4. Repeat this process till the permutation is complete with every element satisfying the S-random criteria.

An S-random interleaver that satisfies the S_a constraint can be generated as follows:

1. Divide the length-N interleaver into $S_a + 1$ bins with $\frac{N}{S_a+1}$ elements placed randomly in each bin.

2. Choose an element from one of the $S_a + 1$ bins. Compute the distance S_b^* between the element and the closest element from the S_a previously chosen elements.

3. Repeat for all n elements by cycling through the $S_a + 1$ bins and choosing an element from each one in succession. S_b is the minimum of all S_b^*.

Lemma 6 *The minimum separation, S_b^*, between the first $S_a^* + 1$ elements chosen for a length-N interleaver is maximized when $S_b^* = \left\lfloor \frac{N-1}{S_a^*} \right\rfloor$.*

Proof. A length-N interleaver is given by

$$\underline{\pi} = \left[\pi_0 \ \pi_1 \ \cdots \ \pi_{N-2} \ \pi_{N-1} \right]$$

where the m^{th} element in the interleaved sequence is the π_m^{th} element of the input sequence. Without loss of generality, assume that the first $S_a^* + 1$ elements of a length-N interleaver are ordered such that $\pi_0 < \pi_1 < ... < \pi_{S_a^*}$. S_b^* is the minimum $(\pi_m - \pi_{m-1})$ for $m \in \left[1, \ldots, S_a^*\right]$. The average separation between successive elements $(\pi_m - \pi_{m-1})$ is given by

$$\frac{\sum_{p=1}^{S_a^*} (\pi_p - \pi_{p-1})}{S_a^*} \leq \frac{N-1}{S_a^*}.$$

S_b^* is maximized when the separation between successive elements $(\pi_m - \pi_{m-1})$ is uniform and equal to the maximum average separation $\frac{N-1}{S_a^*}$. In general,

$$S_{b,\max}^* = \left\lfloor \frac{N-1}{S_a^*} \right\rfloor.$$

■

From Lemma 6, we have that the maximum separation between S_a^*+1 consecutive elements is $S_{b,\max}^* = \left\lfloor \frac{N-1}{S_a^*} \right\rfloor$. Therefore, we can choose at most $S_a^* + 1 = \left\lfloor \frac{N-1}{S_{b,\max}^*} \right\rfloor + 1$ elements in succession and maintain a separation $S_{b,\max}^*$ between each of them. We

will show that for a fixed value of S_b and a length-N interleaver, S_a can be maximized when $N = (S_a + 1)\, S_b$, since

$$\begin{aligned}\left\lfloor \frac{N-1}{S_b} \right\rfloor &= \left\lfloor \frac{(S_a+1)\, S_b - 1}{S_b} \right\rfloor \\ &= \left\lfloor \frac{S_a S_b + S_b - 1}{S_b} \right\rfloor \\ &= \left\lfloor S_a + \frac{S_b - 1}{S_b} \right\rfloor = S_a,\end{aligned}$$

and there exists a systematic method for designing a length-N interleaver such that S_a and S_b satisfy $N = (S_a + 1)\, S_b$:

1. Consider the permutation of the integers $\Big(0 \cdots N-1\Big)$.

2. Partition the integers into $S_a + 1$ bins where

$$\begin{aligned}B_0 &= \{S_b - 1, \ldots 1, 0\} \\ B_1 &= \{2S_b - 1, \ldots, S_b + 1, S_b\} \\ &\vdots \\ B_{S_a} &= \{(S_a+1)\, S_b - 1, \ldots, S_a S_b + 1, S_a S_b\}\end{aligned}$$

3. Choose the first element from the first bin as the first element of the interleaver. Repeat for all S_b bins. Then repeat with the next element from each bin. Cycle through the bins until the interleaver is complete.

If we set $S_a = S_b = S$, we have that

$$S_{\max} = \frac{-1 + \sqrt{1 + 4N}}{2} \approx \sqrt{N}.$$

Example 3 *The following length-*12 *interleaver satisfies* $S_{\max} = \frac{-1+\sqrt{1+4\cdot 12}}{2} = 3$*:*

$$\underline{\pi}_{12} = \Big[\, 2\ 5\ 8\ 11\ 1\ 4\ 7\ 10\ 0\ 3\ 6\ 9 \,\Big].$$

5.1.2 Increasing S for Sub-vector constrained S-random Interleavers

For sub-vector constrained S-random interleavers, the maximum values of the parameters S_a and S_b are functions of the sub-vector length, η, as well as the type of sub-vector interleaver employed. For the case where the sub-vector interleaver is the identity permutation (this is common in the case of sub-vector constrained S-random interleavers for SCCCs), S_a and S_b are jointly maximized when they satisfy the following:

$$S_{a,max} = \frac{N}{m\eta + 1} - 1$$
$$S_{b,max} = m\eta + 1$$

where m is an integer value.

Example 4 *For a length-*15 *sub-vector interleaver employing a length-*4 *identity column permutation, we can choose* $m = 1$ *to permute the integers* $\left(0 \cdots 14\right)$ *written row-wise into a* 4×4 *matrix as follows:*

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & 13 & 14 & - \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 9 & 14 & 3 \\ 8 & 13 & 2 & 7 \\ 12 & 1 & 6 & 11 \\ 0 & 5 & 10 & - \end{bmatrix}.$$

In this example, the S*-parameters can be described as* $S_a = 2$ *and* $S_b = 5$.

The column permutation chosen for the sub-vector interleaver decreases the maximum possible S-parameters. Each case must be analyzed to determine the maximum values of S_a and S_b.

Example 5 *Consider a length-*16 *interleaver employing the following length-*4 *column permutation:*

$$\underline{\pi}_4 = \begin{bmatrix} 2 & 3 & 1 & 0 \end{bmatrix}.$$

We can choose $m = 1$ *to permute the integers* $\left(0 \cdots 15\right)$ *written row-wise into a* 4×4 *matrix as follows:*

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & 13 & 14 & 15 \end{bmatrix} \rightarrow \begin{bmatrix} 2 & 3 & 1 & 0 \\ 6 & 7 & 5 & 4 \\ 10 & 11 & 9 & 8 \\ 14 & 15 & 13 & 12 \end{bmatrix} \rightarrow \begin{bmatrix} 2 & 7 & 9 & 12 \\ 14 & 3 & 5 & 8 \\ 10 & 15 & 1 & 4 \\ 6 & 11 & 13 & 0 \end{bmatrix}.$$

In this example, the S*-parameters can be described as* $S_a = 4$ *and* $S_b = 2$*.*

5.1.3 Sub-vector Constrained S_{max}-random Interleaving

The methods for maximizing S for straight S-random and sub-vector constrained S-random interleavers described in Section 5.1.1 and Section 5.1.2 are not recommended as methods for designing interleavers for turbo codes. Since achieving S_{max} requires the interleaver to be completely constrained (i.e., no random dimension remains in the design technique), we know that interleaver gain will be compromised. We will illustrate this concept with the example of an interleaver designed for the PCCC in Fig. 7.

Example 6 *We will use the column permutation given in (15),*

$$\underline{\pi}_{14} = \begin{bmatrix} 4 & 1 & 11 & 6 & 3 & 5 & 9 & 10 & 13 & 2 & 7 & 12 & 8 & 0 \end{bmatrix},$$

*to create a length-*210 *structured sub-vector interleaver*

$$\underline{\pi}_{210} = \begin{bmatrix} 4 & 15 & 25 & 34 & 45 & 61 & 79 & 94 & 111 & 128 & 147 & 166 & 176 & 196 \\ 18 & 29 & 39 & 48 & 59 & 75 & 93 & 108 & 125 & 142 & 161 & 180 & 190 & 0 \\ 32 & 43 & 53 & 62 & 73 & 89 & 107 & 122 & 139 & 156 & 175 & 194 & 204 & 14 \\ 46 & 57 & 67 & 76 & 87 & 103 & 121 & 136 & 153 & 170 & 189 & 208 & 8 & 28 \\ 60 & 71 & 81 & 90 & 101 & 117 & 135 & 150 & 167 & 184 & 203 & 12 & 22 & 42 \\ 74 & 85 & 95 & 104 & 115 & 131 & 149 & 164 & 181 & 198 & 7 & 26 & 36 & 56 \\ 88 & 99 & 109 & 118 & 129 & 145 & 163 & 178 & 195 & 2 & 21 & 40 & 50 & 70 \\ 102 & 113 & 123 & 132 & 143 & 159 & 177 & 192 & 209 & 16 & 35 & 54 & 64 & 84 \\ 116 & 127 & 137 & 146 & 157 & 173 & 191 & 206 & 13 & 30 & 49 & 68 & 78 & 98 \\ 130 & 141 & 151 & 160 & 171 & 187 & 205 & 10 & 27 & 44 & 63 & 82 & 92 & 112 \\ 144 & 155 & 165 & 174 & 185 & 201 & 9 & 24 & 41 & 58 & 77 & 96 & 106 & 126 \\ 158 & 169 & 179 & 188 & 199 & 5 & 23 & 38 & 55 & 72 & 91 & 110 & 120 & 140 \\ 172 & 183 & 193 & 202 & 3 & 19 & 37 & 52 & 69 & 86 & 105 & 124 & 134 & 154 \\ 186 & 197 & 207 & 6 & 17 & 33 & 51 & 66 & 83 & 100 & 119 & 138 & 148 & 168 \\ 200 & 1 & 11 & 20 & 31 & 47 & 65 & 80 & 97 & 114 & 133 & 152 & 162 & 182 \end{bmatrix}, \quad (38)$$

*for a PCCC employing identical rate-$\frac{1}{2}$ recursive convolutional encoders with three delay states. For the length-*210 *interleaver, $\underline{\pi}_{210}$, $S_a = 11$ and $S_b = 9$.*

We analyzed the weight-distribution of this interleaver used with the SCCC in Fig. 7 and found that there are 183 *free distance codewords with weight* $= 22$.

*We created another length-*210 *sub-vector interleaver with the same column permutation, but with S-random interleaving over each of the columns of the sub-vector interleaver:*

$$\underline{\pi}_{210} = \begin{bmatrix} 4 & 29 & 207 & 20 & 45 & 173 & 107 & 94 & 125 & 184 & 147 & 138 & 8 & 84 \\ 18 & 197 & 39 & 48 & 59 & 75 & 121 & 108 & 97 & 142 & 161 & 152 & 22 & 182 \\ 32 & 43 & 53 & 62 & 73 & 89 & 205 & 122 & 139 & 156 & 21 & 166 & 176 & 196 \\ 46 & 57 & 67 & 76 & 31 & 103 & 93 & 136 & 153 & 170 & 203 & 12 & 190 & 112 \\ 60 & 71 & 81 & 90 & 101 & 33 & 135 & 164 & 181 & 198 & 7 & 54 & 64 & 42 \\ 74 & 85 & 95 & 104 & 115 & 131 & 163 & 150 & 195 & 58 & 175 & 208 & 78 & 0 \\ 88 & 99 & 109 & 118 & 129 & 145 & 51 & 178 & 167 & 30 & 189 & 40 & 204 & 154 \\ 102 & 113 & 123 & 132 & 3 & 61 & 177 & 192 & 83 & 16 & 49 & 26 & 36 & 98 \\ 116 & 127 & 137 & 146 & 157 & 187 & 79 & 206 & 13 & 2 & 35 & 96 & 50 & 70 \\ 130 & 141 & 151 & 160 & 171 & 117 & 191 & 10 & 27 & 44 & 63 & 82 & 106 & 126 \\ 144 & 155 & 165 & 174 & 185 & 201 & 9 & 24 & 41 & 114 & 77 & 68 & 92 & 140 \\ 158 & 169 & 179 & 188 & 199 & 19 & 149 & 38 & 55 & 72 & 91 & 110 & 162 & 28 \\ 172 & 183 & 193 & 202 & 17 & 5 & 37 & 52 & 69 & 86 & 105 & 124 & 134 & 168 \\ 186 & 15 & 25 & 6 & 143 & 159 & 65 & 80 & 111 & 100 & 133 & 180 & 120 & 56 \\ 200 & 1 & 11 & 34 & 87 & 47 & 23 & 66 & 209 & 128 & 119 & 194 & 148 & 14 \end{bmatrix}, \quad (39)$$

In order to make the algorithm for finding the sub-vector constrained S-random simulation run faster, we seeded the interleaver with the permutation in (38). We then modified the sub-vector constrained S-random algorithm to allow random permutations of elements within the interleaver to occur only when they did not decrease the values of S_a and S_b of the original permutation in (38). We allowed this algorithm to iterate until at least $5,000$ *swaps of interleaver elements had occurred.*

The weight distribution analysis of the sub-vector constrained S-random interleaver showed 104 *weight-*22 *free distance codewords. Though the free distance codeword weights are the same for both interleavers, the* multiplicity *of the free distance codewords for the sub-vector constrained S-random interleaver is almost half that of the structured sub-vector interleaver. Fig. 21 shows the error floors due to the BER*

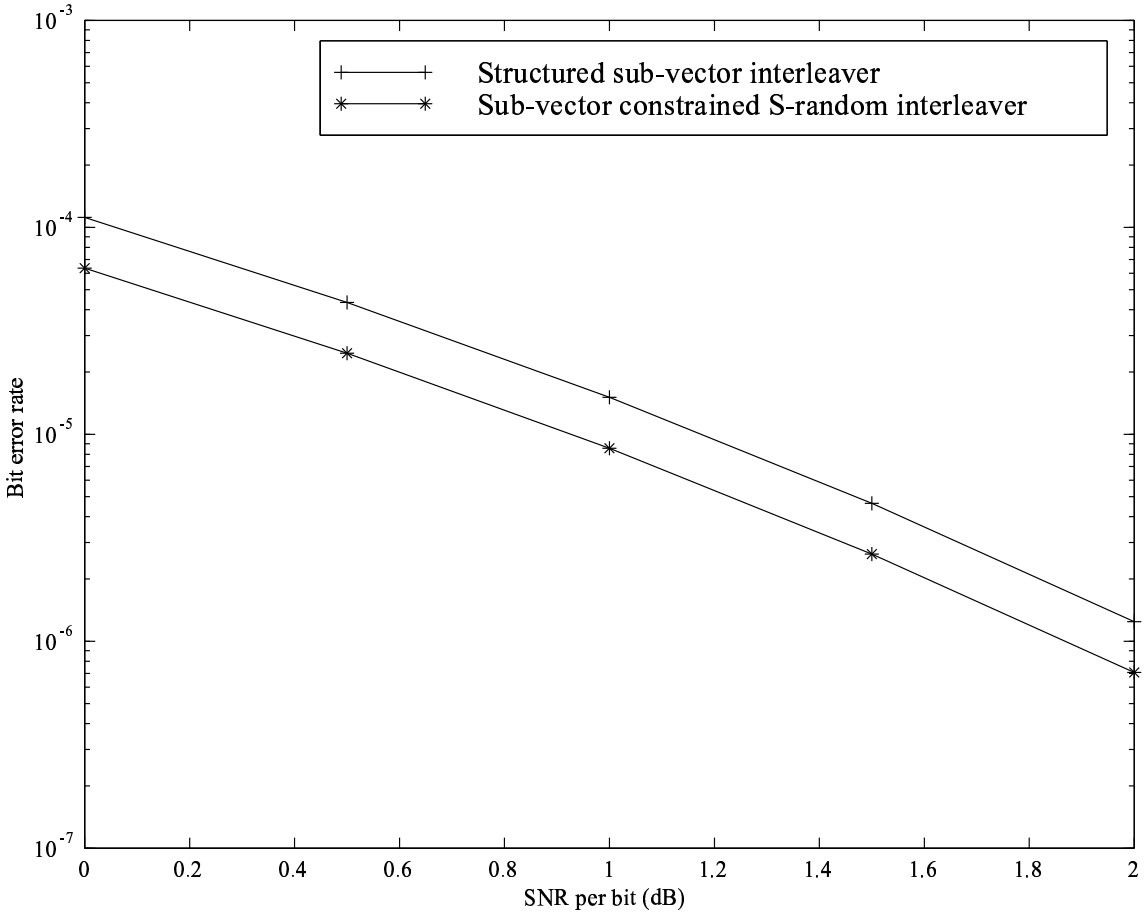

Figure 21: BER contribution of free distance codewords for the length-210 sub-vector constrained S-random interleaver in (39) and the length-210 structured sub-vector interleaver in (38).

contribution of the free distance codewords for both interleavers.

We discussed S_{max}-random interleavers in order to compare the bounds on the design parameter S for S-random and sub-vector constrained S-random interleavers. We then modified the algorithm for creating sub-vector constrained S-random interleavers by allowing them to be seeded with a sub-vector constrained S_{max}-random interleaver. This modification greatly decreases the amount of time necessary to find a sub-vector constrained S-random interleaver for large values of S. We will call this technique "sub-vector constrained S_{max}-random interleaving."

The following is an outline of the method for creating sub-vector constrained S_{max}-random interleavers, valid for length-N sub-vector interleavers where $N \geq \eta^2$:

1. Permute the columns of a matrix with the integers $\{0, 1, \ldots, N-1\}$ read in row-wise using the length-η interleaver

$$\underline{\pi}_\eta = \left[\pi_0 \ \pi_1 \ \cdots \ \pi_{\eta-2} \ \pi_{\eta-1} \right].$$

2. Choose the first η elements of the permutation so that they constitute a set $\{\pi_{0,0}, \pi_{0,1}, \ldots, \pi_{0,\eta}\}$ with increasing values such that $\pi_{0,0} < \pi_{0,1} < \ldots < \pi_{0,\eta}$.

3. If an identity column permutation is used, the spacing between elements can be uniform and equal to $m\eta + 1$, where m is a positive integer, $m \geq 0$. For maximum spacing choose, $m = \left\lfloor \frac{N-1}{\eta} \right\rfloor$. For non-identity column permutations, the spacing between elements will not be uniform.

4. For subsequent rows of the interleaver, choose $\pi_{a,b} = \pi_{a-1,b} + \eta$. This initial permutation is used to seed the sub-vector constrained S-random algorithm. Analyze the interleaver to determine the values of S_a and S_b.[2]

5. Beginning with the first element in the interleaver, $\pi_{0,0}$, randomly select a different element from the same column of the interleaver, $\pi_{a,0}$, to swap positions with $\pi_{0,0}$:

$$\pi_{a,0} \in \left\{ \pi_{0,1}, \pi_{0,2}, \pi_{0,3}, \ldots, \pi_{0,N/\eta-1} \right\}.$$

6. If distance between the new values of $\pi_{0,0}$ and $\pi_{a,0}$ and the S_a previously selected elements is $\geq S_b$, set $\underline{\pi}_N(0,0) = \pi_{a,0}$ and $\underline{\pi}_N(a,0) = \pi_{0,0}$. Otherwise, do not allow the swap.

[2]This can be done manually for short interleavers, or by computer search for longer interleavers. Possible values of S_a and S_b would be chosen with the length of the critical low-weight codewords for the PCCC or SCCC in mind.

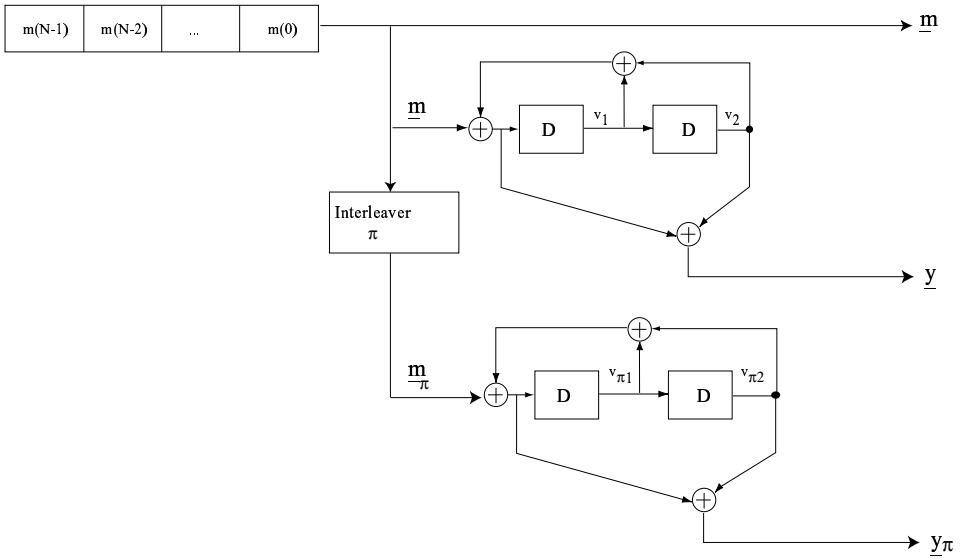

Figure 22: A diagram of a PCCC employing identical rate-$\frac{1}{2}$ recursive convolutional encoders with two delay states.

7. Proceed row-wise through the sub-vector interleaved sequence until a predetermined number of swaps have been performed successfully. The interleaver is complete with every element satisfying the sub-vector constrained S-random criteria.

5.2 Sub-vector Constrained S-random Interleaving for PCCCs

We designed a length-162 sub-vector constrained S-random interleaver for a PCCC shown in Fig. (22) employing identical two delay-state recursive convolutional encoders. We chose a length-6 interleaver to permute the columns of the sub-vector interleaver,

$$\underline{\pi}_6 = \left[0\ 1\ 2\ 5\ 3\ 4 \right],$$

since this length corresponds to two periods of the feedback polynomial of the constituent encoders. Therefore, it is possible to use the length-6 sub-vector column interleaver to break up the shortest weight-2 terminating input sequences for these

encoders:

$$\underline{\pi}_6\left(\left[\begin{array}{cccccc}1&0&0&1&0&0\end{array}\right]\right) \rightarrow \left[\begin{array}{cccccc}1&0&0&0&1&0\end{array}\right]$$
$$\underline{\pi}_6\left(\left[\begin{array}{cccccc}0&1&0&0&1&0\end{array}\right]\right) \rightarrow \left[\begin{array}{cccccc}0&1&0&0&0&1\end{array}\right]$$
$$\underline{\pi}_6\left(\left[\begin{array}{cccccc}0&0&1&0&0&1\end{array}\right]\right) \rightarrow \left[\begin{array}{cccccc}0&0&1&1&0&0\end{array}\right]$$

However, not all cyclic shifts of the weight-3 terminating input sequence that generates the weight-5 free distance codeword for the constituent encoders are interleaved into non-terminating input sequences by this sub-vector interleaver:

$$\underline{\pi}_6\left(\left[\begin{array}{cccccc}1&1&1&0&0&0\end{array}\right]\right) \rightarrow \left[\begin{array}{cccccc}1&1&1&0&0&0\end{array}\right]$$
$$\underline{\pi}_6\left(\left[\begin{array}{cccccc}0&0&0&1&1&1\end{array}\right]\right) \rightarrow \left[\begin{array}{cccccc}0&0&0&1&1&1\end{array}\right]$$

In fact, an exhaustive search of length-6 interleavers reveals none that can achieve this goal. Therefore, the S-random interleaving performed over each column of the sub-vector interleaver is what increases the minimum free distance codeword weight of this coding scheme. For the length-162 sub-vector constrained S-random interleaver, we chose $S = 7$, since this spread is sufficient to break-up the terminating input sequences that generate the free distance codewords for the PCCC.

The BER performance of this PCCC employing length-160 interleavers designed by three different short interleaver design methods was simulated by Daneshgaran and Mondin and reported in [15]. To verify the performance of our method of sub-vector constrained S-random interleaving, we repeated the simulations published in [15] of a length-160 straight S-random interleaver (again, $S = 7$) and a length-160 interleaver designed by the method described in [15]. We compared them to the simulation of our sub-vector constrained S-random interleaver and found that sub-vector constrained S-random interleaving outperforms all of these methods at the SNRs we simulated.

We computed the weight distribution of this encoding scheme using the method described in [22]. The lower bound on the BER suggested by the weight distribution

Table 8: Weight distribution up to weight 21 of PCCC in Fig. 22 employing three different types of interleavers. Interleaver lengths are approximately 160, d is the codeword weight, w is the input sequence weight, and *mult* is the codeword mutiplicity.

		S-random	*Daneshgaran and Mondin [15]*	*Sub-vector Constrained S-random*
d	w	*mult*	*mult*	*mult*
14	2	7	–	–
15	2	1	–	–
16	2	10	–	–
17	2	1	–	–
	3	–	4	–
18	2	9	14	17
	3	1	–	–
19	2	1	–	–
	3	4	18	5
20	2	7	26	–
	3	2	–	–
	4	10	5	1
	6	–	4	–
21	2	2	–	–
	3	6	33	8
	5	–	–	5

is also lower than the bounds of the other methods reported in [15]. These lower bounds are a "truncated" union bound on the PCCC which is a sum of the BER contribution of the lowest weight codewords which was computed using data from the weight distribution analysis presented in Table 8. Results of these simulations and their lower bounds are shown in Fig. 23.

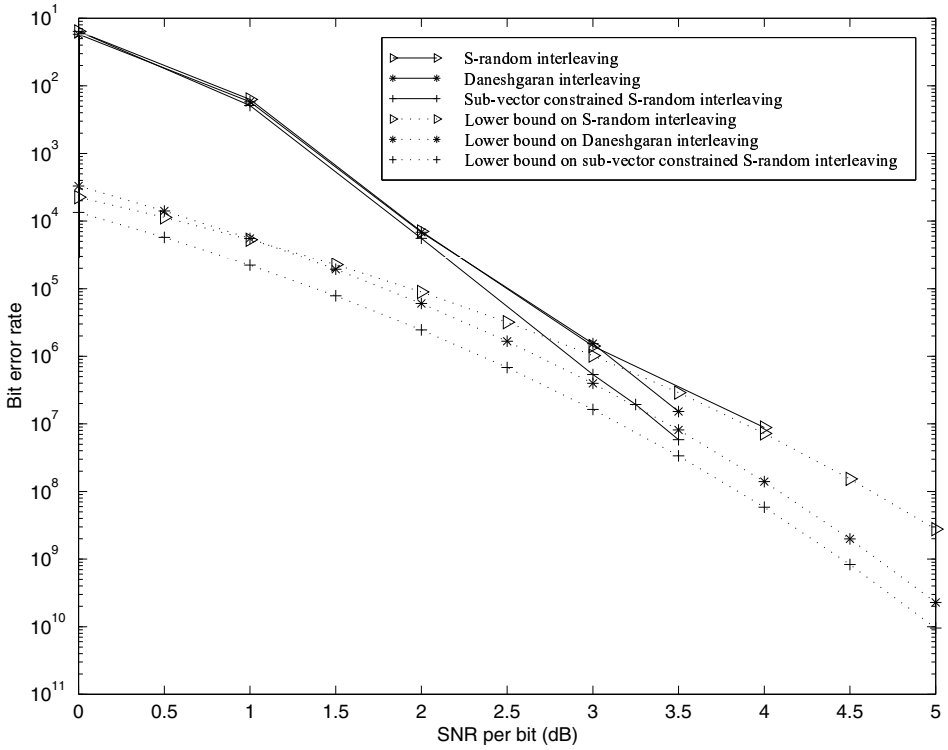

Figure 23: Simulated BER plot for a PCCC employing identical two delay-state encoders and interleavers designed by three different methods. The interleaver lengths are ≈ 160 and the simulated BER is plotted along with its lower bound.

5.3 Sub-vector Constrained S-random Interleaving for SCCCs

Incorporating concepts from S-random interleaving into the sub-vector interleaving method leads to an improvement on both of these methods of interleaver design, as it did in the case of PCCCs. The algorithm for incorporating S-random concepts into sub-vector interleavers for SCCCs is the same as for PCCCs.

Distance spectrum analysis shows that the average free distance codeword weight is increased with sub-vector constrained S-random interleaving compared to straight S-random interleaving.

5.3.1 Distance Spectrum

As in Section 4.5.4, we were able to compute expected multiplicity for codewords of the SCCC in Fig. 12 as a function of the codeword's Hamming weight. The results of this analysis are presented in Table 9 which lists the expected input sequence weight, $E[w]$, and expected multiplicity, $E[mult]$, of codewords with Hamming weight ≤ 20 assuming a length-84 average S-random and a length-84 average sub-vector constrained S-random interleaver.

In Fig. 24 we plot the BER contribution of error patterns corresponding to codewords with Hamming weight $d \leq 9$. Our weight distribution analysis shows us that the BER contribution of the codewords with Hamming weight $d \leq 8$ can be eliminated by sub-vector interleaving, and the multiplicity of other low Hamming weight codewords is reduced.

Results of a Monte Carlo simulation of this SCCC which confirms this analysis will be shown in Sec. 5.3.2.

5.3.2 Simulation Results

We simulated the performance of the SCCC in Fig. 15 employing both a length-84 S-random interleaver and a length-84 sub-vector constrained S-random interleaver. We

Table 9: Weight distribution up to weight 20 of SCCC in Fig. 12 employing an average length-84 S-random and an average length-84 sub-vector constrained S-random interleaver. The codeword weight, d, expected input sequence weight, $E[w]$, and expected multiplicity, $E[mult]$, was computed from an ensemble of $100,000$ randomly generated interleavers.

	$S-random$		*Sub-vector constrained S-random*	
d	$E[w]$	$E[mult]$	$E[w]$	$E[mult]$
6	5.00	$\frac{4}{10000}$	–	–
7	3.00	$\frac{386}{10000}$	–	–
8	4.31	$\frac{72}{10000}$	–	–
9	3.31	$\frac{408}{10000}$	3.98	$\frac{98}{10000}$
10	3.57	$\frac{614}{10000}$	4.63	$\frac{82}{10000}$
11	2.54	$\frac{1834}{10000}$	4.12	$\frac{360}{10000}$
12	3.62	$\frac{1186}{10000}$	4.83	$\frac{384}{10000}$
13	3.02	$\frac{2662}{10000}$	4.93	$\frac{1737}{10000}$
14	4.24	$\frac{4204}{10000}$	5.30	$\frac{2377}{10000}$
15	3.68	$\frac{9580}{10000}$	5.10	$\frac{6030}{10000}$
16	4.22	$\frac{11520}{10000}$	5.88	$\frac{6480}{10000}$
17	4.72	$\frac{15210}{10000}$	5.88	$\frac{15930}{10000}$
18	5.18	$\frac{24810}{10000}$	6.33	$\frac{22350}{10000}$
19	5.08	$\frac{38650}{10000}$	6.35	$\frac{43100}{10000}$
20	5.64	$\frac{61020}{10000}$	7.23	$\frac{60580}{10000}$

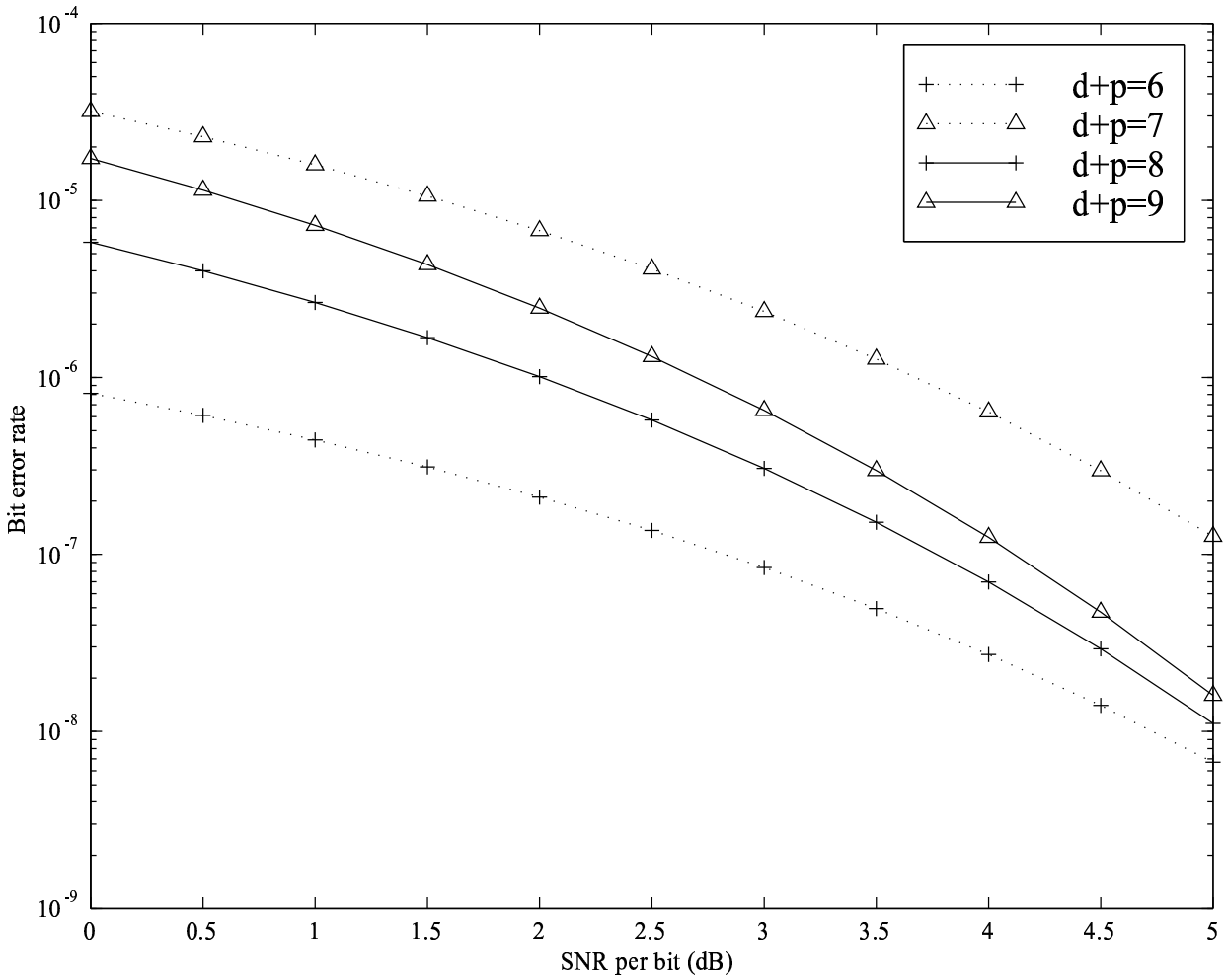

Figure 24: BER contribution of error patterns corresponding to codewords with Hamming weight $d \leq 9$ for the SCCC in Fig. 12 and a length-84 S-random interleaver. Sub-vector interleaving eliminates the BER contribution of the error patterns corresponding to codewords with Hamming weight $d \leq 8$.

Table 10: Weight-3 and weight-4 codewords from the outer encoder of the SCCC in Fig. 12.

Weight-3 Codewords	Weight-4 Codewords
$\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$
$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$
	$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$
	$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$
	$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

chose a length-7 identity permutation for the columns of our sub-vector interleaver,

$$\underline{\pi}_7 = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{bmatrix},$$

since the low weight codewords from the outer encoder do not correspond to terminating input sequences for the inner encoder. Table 10 lists the weight-3 and weight-4 codewords from the outer encoder.

The free distance codeword for the length-84 S-random interleaved SCCC in Fig. 12 is expected to occur in $\frac{4}{10000}$ randomly generated S-random interleavers.[3] This codeword is generated by a weight-4 input sequence, $\underline{m}$, to the outer encoder:

$$\underline{m} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

The input sequence $\underline{m}$ generates the parity sequence output, $\underline{p}$, from the outer encoder:

$$\underline{p} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

The parity sequence output, $\underline{p}$, can be interleaved by an S-random interleaver $\underline{\pi}$ that satisfies the constraint $S = 3$ as follows:

$$\underline{\pi}\left(\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}\right) \rightarrow \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

[3]Here we used $S = 3$, a low value for the spread, so that our simulations would generate errors more quickly.

This interleaved sequence $\underline{p}_\pi$ generates the following weight-2 parity sequence output from the outer encoder:

$$\left[1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\right].$$

The result is the weight-6 free distance codeword for the SCCC employing an S-random interleaver.

The free distance codeword for the length-84 sub-vector constrained length-84 S-random interleaved SCCC in Fig. 12 is expected to occur in $\frac{98}{10000}$ randomly generated sub-vector constrained S-random interleavers.[4] This codeword is generated by a weight-5 input sequence, $\underline{m}$, to the outer encoder:

$$\underline{m} = \left[1\ 1\ 0\ 1\ 1\ 1\ 0\ 0\right].$$

The input sequence $\underline{m}$ generates the parity sequence output, $\underline{p}$, from the outer encoder:

$$\underline{p} = \left[1\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\right].$$

The parity sequence output, $\underline{p}$, can be interleaved by a sub-vector constrained S-random interleaver $\underline{\pi}$ that satisfies the constraint $S = 3$ as follows:

$$\underline{\pi}\left(\left[1\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\right]\right) \rightarrow \left[1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\right]$$

This interleaved sequence $\underline{p}_\pi$ generates the following weight-2 parity sequence output from the outer encoder:

$$\left[1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\right].$$

The result is the weight-9 free distance codeword for this SCCC employing a length-84 sub-vector constrained S-random interleaver.

We computed the weight distribution of this encoding scheme using the method described in [22] for the SCCC employing both average length-84 S-random and

[4] Again, we used $S = 3$

sub-vector constrained S-random interleavers (as before, using $S = 3$). The weight distribution of codewords up to weight-20 is presented in Table 9. The codeword weight, d, input sequence weight, w, and expected multiplicity, m, was computed from an ensemble of $100,000$ randomly generated interleavers in both cases.

The BER performance of this SCCC in Fig. 12 employing length-84 S-random and sub-vector constrained S-random interleavers is presented in Fig. 25. Lower bounds on the BER are plotted beneath the performance curves. These lower bounds are a "truncated" union bound on the SCCC which is a sum of the BER contribution of codewords less than weight-24 which were computed from the weight distribution analysis (partial results of this analysis are presented in Table 9).

5.3.3 Average Free Distance

Using the same method used to compute the weight distribution of the SCCC, we computed the average free distance codeword weight for the SCCC in Fig. 16 employing both an S-random interleaver and a sub-vector constrained S-random interleaver as a function of interleaver length. We punctured the parity sequence output from the inner encoder so that the overall rate of the SCCC was $\frac{1}{3}$. We see in Fig. 26 that the average free distance codeword weight for the sub-vector constrained S-random interleaver is greater than that of the S-random interleaver. The difference between the average free distance codeword weights also increases with interleaver length.

5.4 Summary

For short interleavers in particular, sub-vector constrained S-random row interleaving does a better job than random row interleaving for breaking the low input weight-2 and weight-3 terminated codewords.

Results presented in Section 3.4.2 of the BER simulation of a PCCC with a length-162 interleaver and two delay-state recursive convolutional encoders suggest that sub-vector constrained S-random interleavers perform better than some of the

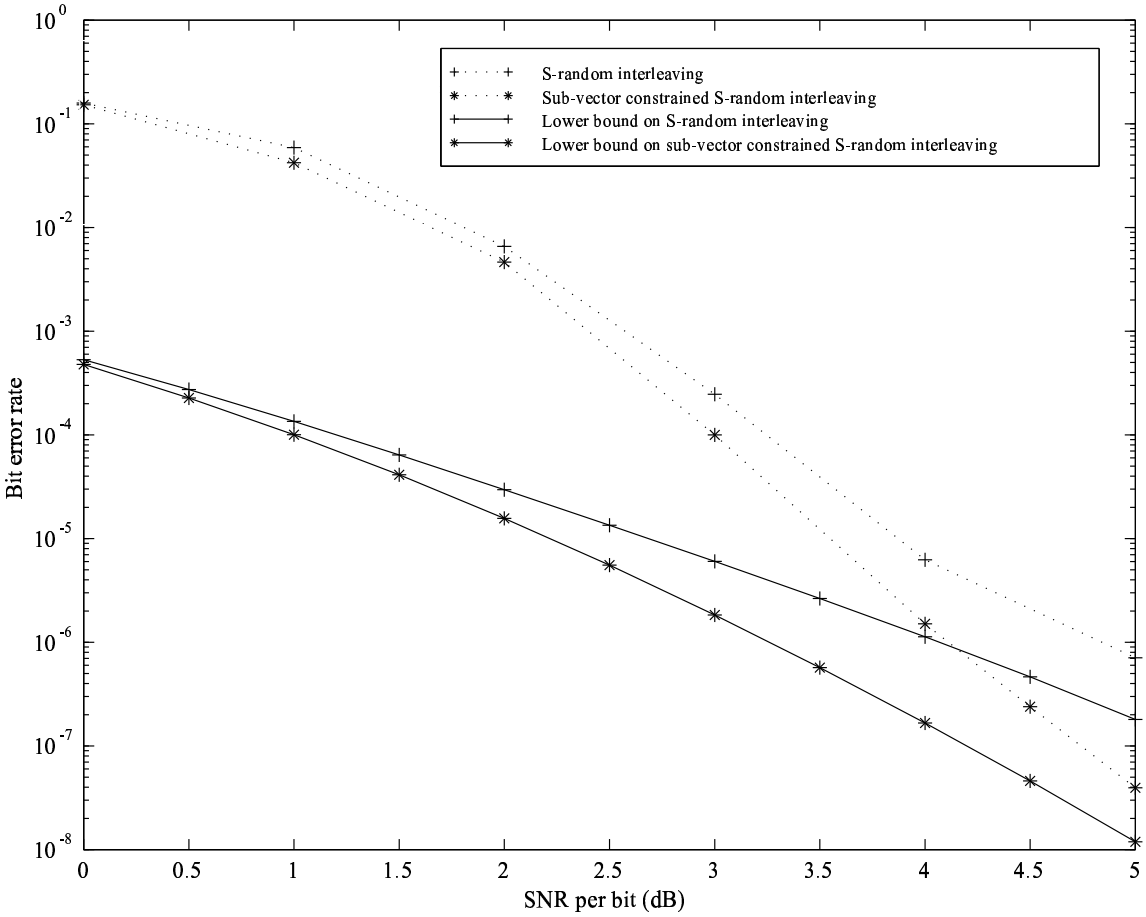

Figure 25: Simulated BER of the SCCC in Fig. 12 employing a length-84 sub-vector constrained S-random interleaver versus a length-84 S-random interleaver.

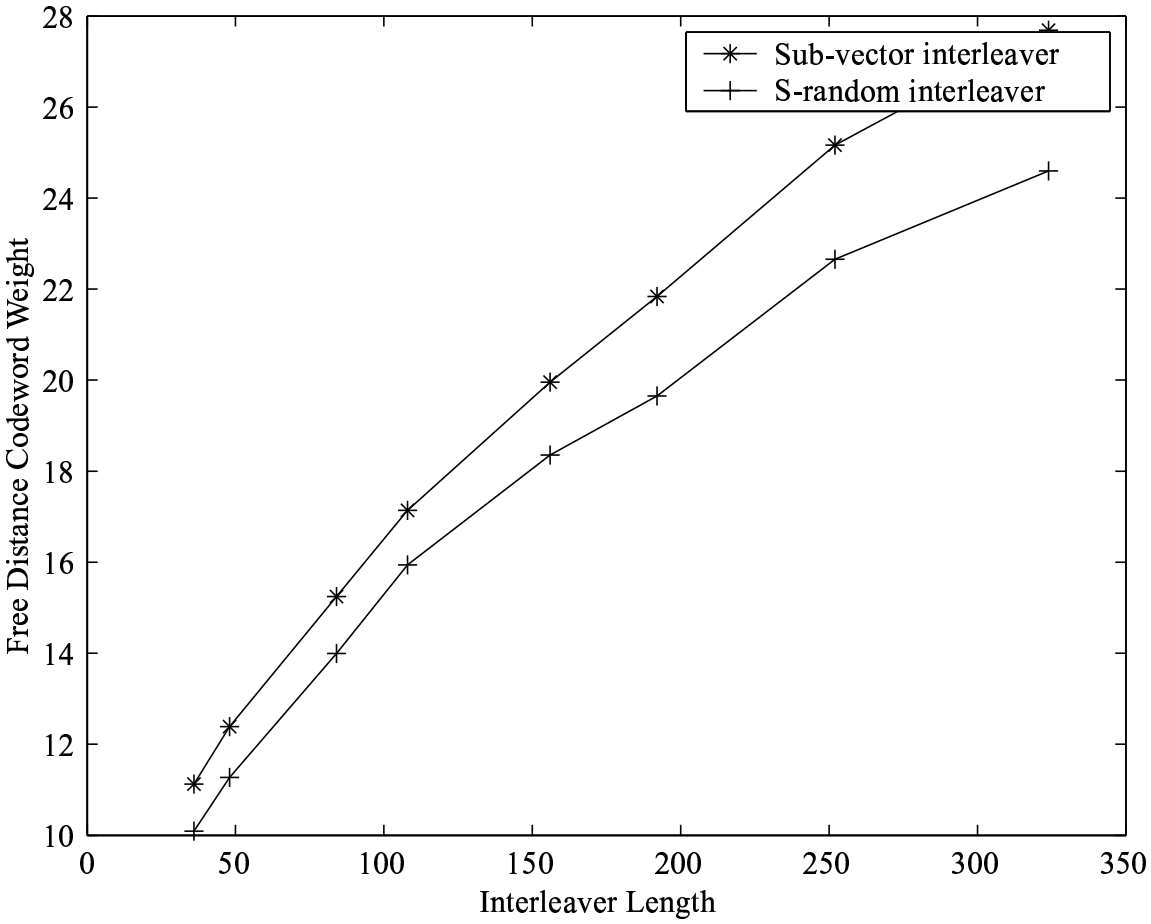

Figure 26: Average free distance of the SCCC in Fig. 16, with the inner encoder parity sequence punctured so that the overall rate of the coding scheme is $\frac{1}{3}$, employing a length-N sub-vector constrained S-random interleaver versus a length-N S-random interleaver. $S = \sqrt{\frac{N}{2}}$.

best methods known for designing short interleavers for PCCCs.

The contributions of the research described in this chapter are as follows:

- A method for computing the maximum spread, S_{max}, possible for a length-N S-random interleaver.
- A systematic method for choosing S-random interleavers that satisfy $S = S_{max}$.
- Sub-vector constrained S-random interleaving for PCCCs and SCCCs, a method of interleaving of the same complexity as S-random interleaving, but with better BER performance in the error floor.

CHAPTER VI

DECODER CONVERGENCE

Simulations in previous chapters of PCCCs and SCCCs employing sub-vector interleavers show improvement over the average random (in Fig. 11 and Fig. 19) and S-random (in Fig. 23 and Fig. 25) interleavers. However, these simulations do not always show convergence to the lower bounds computed analytically, especially in the case of coding schemes employing short interleavers and at low SNRs.

The turbo code decoding algorithm is described as a discrete dynamical system in [1], iterating on the extrinsic information output from the constituent decoders. As a result, the modes of convergence of the decoding algorithm can be described in terms of its properties as a dynamical system, as in [38].

These modes of convergence are as follows:

1. *Convergence to an unequivocal fixed point:* Most decisions by the decoder correspond to maximum-likelihood decoding of the received sequence. Errors that occur when the decoder has converged to an unequivocal fixed point typically correspond to low weight codewords for the turbo coding scheme.

2. *Convergence to an indecisive fixed point:* The extrinsic information passed between the two decoders remains very low, indicating that the decoding algorithm is ambiguous regarding the values of the information bits. These unequivocal fixed points can correspond to a large number of bit errors in the decoded sequence.

3. *Convergence to an invariant set:* This generally occurs in the "waterfall region" of the range of SNRs and occurs when an indecisive fixed point bifurcates. In

practice this is observed as the decoder converging on a set of fixed points and oscillating among them periodically.

Sub-vector interleavers are designed to increase the free distance weight of codewords in a PCCC or SCCC, and also to decrease the multiplicity of some low-weight codewords in a PCCC or SCCC. Improvement to the BER of a PCCC or SCCC can be attributed to the impact of the sub-vector interleaver on the first mode of convergence, convergence to an unequivocal fixed point. Maximum-likelihood decoding of an error pattern will result in the decoding of the error pattern as a codeword of the PCCC or SCCC. Elimination of certain low-weight or high-multiplicity codewords improves the performance of the coding scheme in the error floor region where the decoder typically converges to an unequivocal fixed point corresponding to a maximum-likelihood decision on the decoded sequence. However, sub-vector interleaving does not affect the bit error rates when the second two modes of convergence, convergence to an indecisive fixed point and convergence to an invariant set, prevail. At low SNRs, these two modes of convergence dominate the performance of the coding scheme and improvement in BER performance due to sub-vector interleaving is difficult to detect.

In this section, the results of simulations of PCCCs and SCCCs at SNRs in the waterfall region and below will be shown where the mode of convergence was determined to be either to an unequivocal fixed point, an indecisive fixed point, or an invariant set. By classifying decoding errors according to the mode of convergence of the decoder, the predicted improvements in BER performance of coding schemes employing sub-vector interleaving can be observed, even in regions where Monte Carlo simulations do not show convergence of the BER to the lower bounds computed analytically.

6.1 Determining the Mode of Convergence of the Decoder

We used a variation of a method for determining when decoding should be terminated proposed in [38], which was a hybrid of methods proposed in [23] and [41], to classify decoded frames according to the mode of convergence of the decoder. The goal was to extract the errors that occurred when the decoder converged to a maximum-likelihood decision on the input sequence. By doing this, we were able show improvement in the BER performance of the coding scheme when employing sub-vector interleavers. Each decoded frame was classified as corresponding to an indecisive fixed point, an invariant set, or an unequivocal fixed point by the following steps:

1. *Indecisive fixed point:* In order to identify errors that occurred when the decoder converged on an indecisive fixed point, we determined a threshold on the average extrinsic information passed between the two decoders. If the average magnitude of the extrinsic information did not exceed this threshold, convergence to an indecisive fixed point was declared for the frame. The threshold was obtained by performing a trial run of the decoder at each SNR to be simulated where the average value of the magnitude of the extrinsic information, μ_e, of 1000 correctly decoded frames was calculated. The variance, σ_e^2 was also calculated, and a threshold set at $\mu_e - 1.5 \cdot \sigma_e$.[1]

2. *Invariant set:* The sign of the extrinsic information associated with each decoded bit was determined after each iteration of the decoder. If the threshold on average extrinsic information was met, but the number of sign changes between each iteration did not equal zero before a predetermined maximum number of iterations, convergence to an invariant set was declared for the frame.[2]

[1] In practice this variance was very small due to the residual values added to the extrinsic information at each decoding iteration to prevent numerical overflow.

[2] If the extrinsic information > 0, this implies that a hard decision on the decoded sequence $= 0$, based on the extrinsic information. If the extrinsic information < 0, this implies that a hard decision

3. *Unequivocal fixed point:* Decoding errors that were not determined to have occurred as a result of decoder convergence to an indecisive fixed point or an invariant set were assumed to have occurred as a result of decoder convergence to a maximum-likelihood decision on the encoded sequence.

Fig. 27 shows the simulated BER performance of the SCCC shown in Fig. 16 employing both length-120 S-random and sub-vector constrained S-random interleavers. This particular SCCC was chosen for simulation because the fact that it is associated with relatively low-weight codewords (compared to a SCCC with more delay states or a more powerful outer encoder) should lead to better convergence properties in the decoder at low SNRs, as described in [2]. These results are plotted along with a lower bound on the BER (i.e., the free distance codeword asymptote). We see that the BER performance has not come close to approaching the error floor at the SNRs used in the simulation. We have found that the lack of convergence of the BER performance at low SNRs to the lower bounds computed analytically is particularly noticeable in SCCCs. Furthermore, SCCCs have lower error floors with steeper slopes than PCCCs and, at SNRs where we might expect to find convergence to the error floor in our simulations, the error rates are lower than we can attempt to simulate!

6.2 Analysis of SCCCs Employing Very Short Interleavers

Simulations of the SCCC shown in Fig. 16 were performed with length-24 and length-36 sub-vector interleavers. The free distance codeword of the outer encoder is generated by the input sequence

$$\underline{m} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$$

on the decoded sequence = 1, based on the extrinsic information.

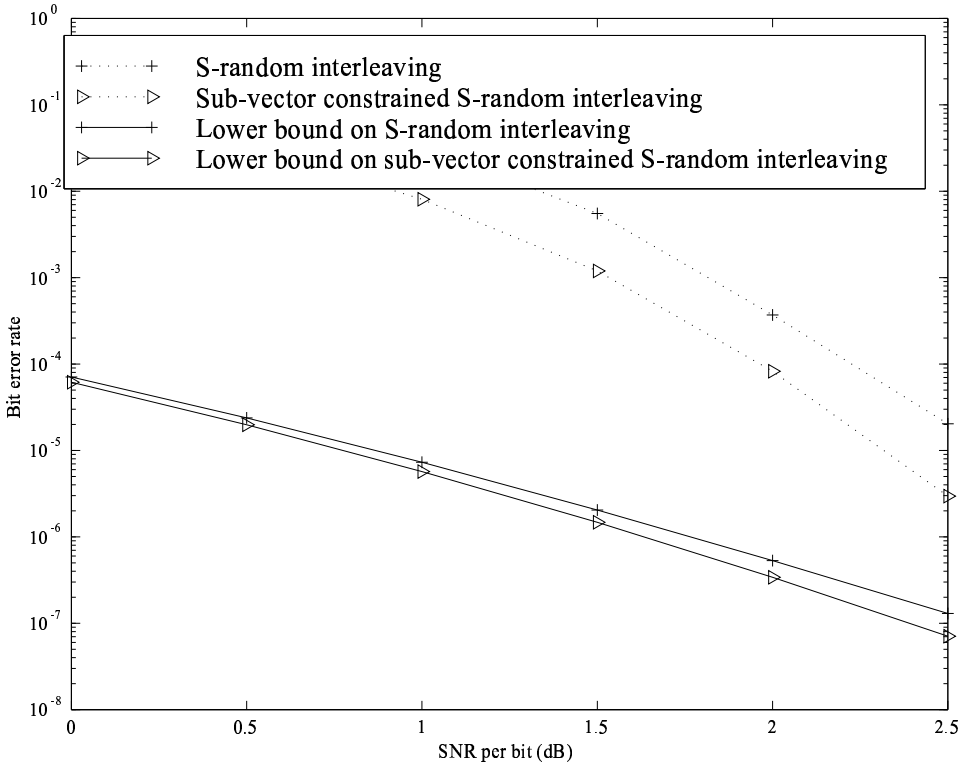

Figure 27: Simulated BER performance of the SCCC shown in Fig. 16 employing both length-120 S-random and sub-vector constrained S-random interleavers. The free distance codeword asymptotes are plotted below the performance curves.

that combines with the parity sequence output from the outer encoder to create the weight-5 free distance codeword

$$\underline{p} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

All length-6 cyclic shifts of the free distance codeword, as well as the weight-6 codewords, generated by the outer encoder can be interleaved into a non-terminating input sequence for the inner encoder by a length-6 identity interleaver

$$\underline{\pi}_6 = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \end{bmatrix},$$

used to permute the columns of our sub-vector interleaver. As in shown (25), interleaving the weight-d_{free} and weight-$d_{free}+1$ codewords into non-terminating input sequences to the inner encoder increases the interleaver gain of the SCCC by a factor of $\left(\frac{1}{N}\right)$.

The simulated converged BER of the SCCC in Fig. 16 employing both a length-24 and a length-36 sub-vector interleaver is also shown in Fig. 28. The converged BER is the BER computed for the frames that have been determined to correspond to unequivocal fixed points of the decoder by the steps outline above. The converged BER is compared to the BER computed when including frames that correspond to all modes of decoder convergence in Fig. 28. The converged BER shows a significant improvement over the all-inclusive BER. However, we see in Fig. 29 that the overall rate of the coding scheme is reduced by almost 10% at moderate SNRs, up to more than 50% when the SNR is 0dB.

Length-24 and length-36 sub-vector interleavers were chosen for simulation because their short lengths made them easier to analyze:

- It was easier (i.e., faster) to generate a statistically significant number of errors to accurately estimate the BER in the error floor region of SCCCs employing short interleavers rather than longer interleavers because of the relatively high error floors present when using the short interleavers.

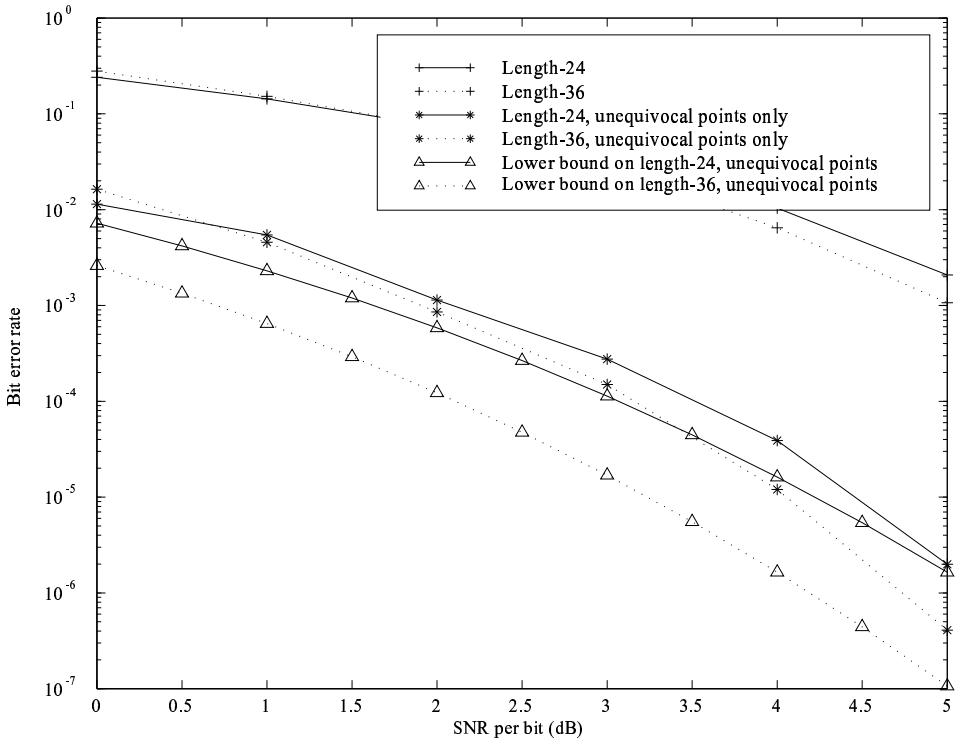

Figure 28: Simulated BER and converged BER plots (i.e., BER when decoder has converged to unequivocal fixed points) for an SCCC employing the two identical rate $\frac{1}{2}$ four state encoders shown in Fig. 16 and both a length-24 and a length-36 sub-vector interleaver.

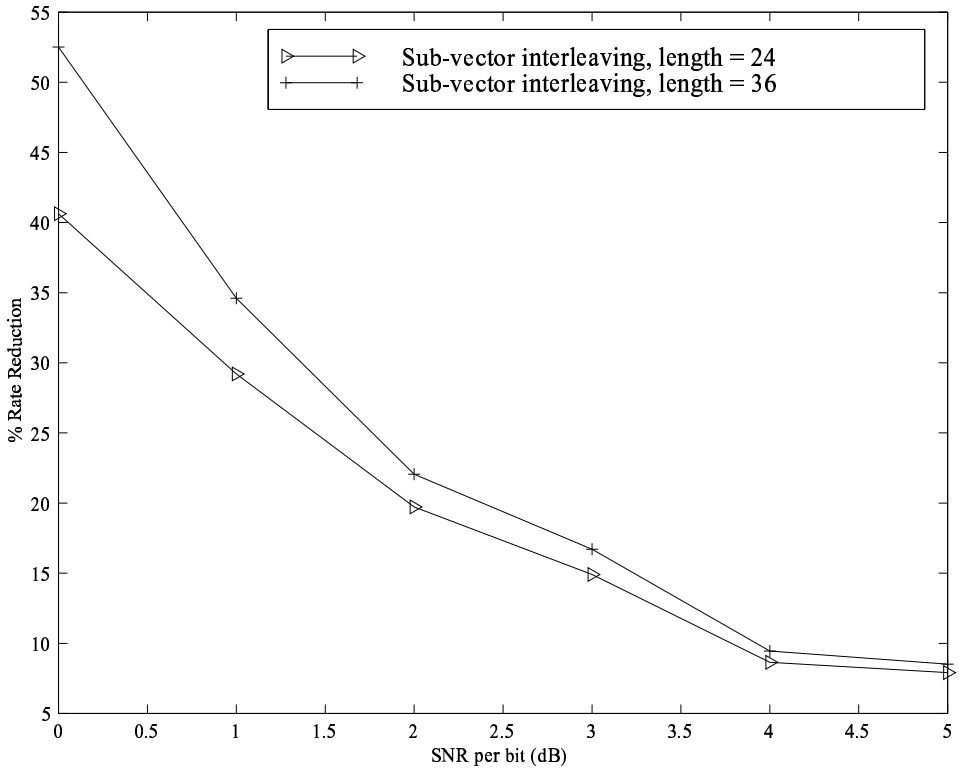

Figure 29: Rate reduction in the simulation of the BER of the SCCC shown in Fig. 16 caused by rejecting frames determined to have converged to indecisive fixed points or invariant sets of the decoder.

- It was feasible to compute the union bound for the SCCC employing short interleavers.

The union bound for the SCCCs was obtained using the method outlined in [22] to compute the weight distribution of all codewords for the SCCC. An average weight distribution of length-24 and length-36 interleavers was calculated by taking the average of $100,000$ weight distributions of randomly generated length-24 and length-36 sub-vector interleavers, respectively. Partial results, up to weight-48 codewords, of the average weight distributions are presented in Table 14 and Table 15.

A lower bound on the sub-vector interleaved SCCCs was obtained by computing the weight distribution of all codewords generated by a weight-7 input sequence to the inner encoder. The free distance codeword of this particular SCCC was generated by a weight-7 input sequence

$$\underline{p}_{\pi} = \left[1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1 \right]$$

to the inner encoder that generated a weight-2 parity sequence output

$$\underline{y} = \left[1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \right]$$

from the inner encoder, for an overall weight-9 free distance codeword. The high multiplicity codeword (i.e., $d_{free,eff}$) for this SCCC was generated by the same weight-7 codeword from the outer encoder as the free distance codeword (see Table 12) interleaved into three separate terminating input sequences to form the input sequence

$$\underline{p}_{\pi} = \left[1\ 1\ 1 \cdots 1\ 0\ 0\ 1 \cdots 1\ 0\ 0\ 1 \right]$$

to the inner encoder that generate a weight-10 parity sequence output

$$\underline{y} = \left[1\ 0\ 1 \cdots 1\ 1\ 1\ 1 \cdots 1\ 1\ 1\ 1 \right]$$

from the inner encoder, for an overall weight-17 effective free distance codeword.

The probability of a bit error due to an error pattern corresponding to a weight-$d_{free} = 9$ codeword is computed using the data in Table 14:

$$P_b\left(e \mid \underline{e}_{w,d_{free}}\right) \approx \left(\frac{E\left[w|d = d_{free}\right]}{N_{out}}\right) \cdot E\left[mult|d = d_{free}\right] \cdot Q\left(\sqrt{(d_{free}) \cdot \frac{2R_c\varepsilon_b}{N_0}}\right)$$
$$\approx \left(\frac{3.89}{25}\right) \cdot \left(\frac{23}{10,000}\right) \cdot Q\left(\sqrt{(9) \cdot \frac{2R_c\varepsilon_b}{N_0}}\right), \text{ for the length-24 sub-vector interleaver}$$
$$\approx \left(\frac{3.86}{36}\right) \cdot \left(\frac{4}{10,000}\right) \cdot Q\left(\sqrt{(9) \cdot \frac{2R_c\varepsilon_b}{N_0}}\right), \text{ for the length-36 sub-vector interleaver}$$

The probability of a bit error due to an error pattern corresponding to a weight-$d_{free,eff} = 17$ codeword is computed using the data in Table 14:

$$P_b\left(e \mid \underline{e}_{w,d_{free}}\right) \approx \left(\frac{E\left[w|d = d_{free}\right]}{N_{out}}\right) \cdot E\left[mult|d = d_{free}\right] \cdot Q\left(\sqrt{(d_{free}) \cdot \frac{2R_c\varepsilon_b}{N_0}}\right)$$
$$\approx \left(\frac{4.06}{24}\right) \cdot \left(\frac{50197}{10,000}\right) \cdot Q\left(\sqrt{(17) \cdot \frac{2R_c\varepsilon_b}{N_0}}\right), \text{ for the length-24 sub-vector interleaver}$$
$$\approx \left(\frac{4.00}{36}\right) \cdot \left(\frac{16143}{10,000}\right) \cdot Q\left(\sqrt{(17) \cdot \frac{2R_c\varepsilon_b}{N_0}}\right), \text{ for the length-36 sub-vector interleaver}$$

These two error probabilities form an asymptotic lower bound on the BER and are plotted as lower bounds in Fig. 30 and Fig. 31.

We computed the total weight distribution for average length-24 and length-36 sub-vector interleavers by averaging the weight distributions of $100,000$ randomly generated sub-vector interleavers. Using these average weight distributions, we computed the union bounds for the SCCC employing average length-24 and length-36 sub-vector interleavers. We see in Fig. 30 and Fig. 31 that the simulated converged BER performance is bounded above and below by these two bounds at low SNRs, and approaches the lower bound at moderate and high SNRs.

6.2.1 Interleaver Gain

Using the weight distributions computed for length-24 and length-36 average random and average sub-vector interleavers (see Tables 15 and 14), we plotted the union bound on the BER for the SCCC in Fig. 16 employing these four types of interleavers

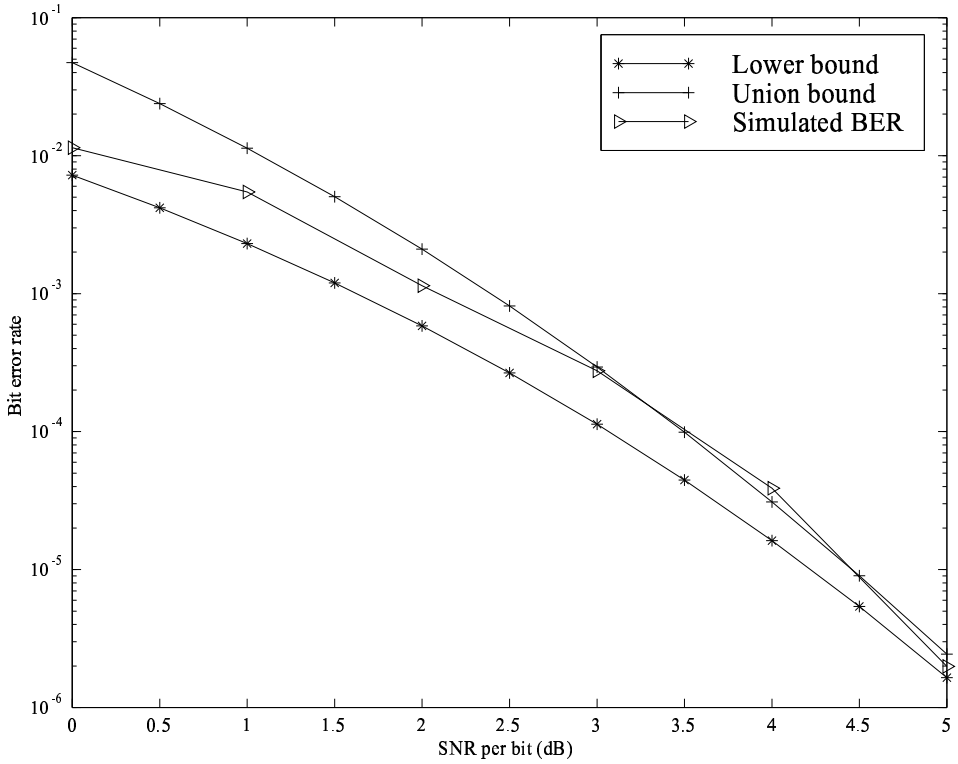

Figure 30: Simulated converged BER of the SCCC shown in Fig. 16 employing a length-24 sub-vector interleaver plotted along with the union bound and a lower bound on the SCCC.

Table 11: A table of outer encoder terminating input sequences and terminated output sequences that generate weight-7 terminating input sequences to the inner encoder.

Input Sequence, $\underline{m}$		Output Sequence, $\underline{y}$
$[1\ 0\ 1\ 0\ 1\ 0\ 0]$	$\rightarrow$	$[1\ 1\ 0\ 1\ 1\ 0\ 0]$
$[1\ 1\ 0\ 1\ 0\ 1\ 1]$	$\rightarrow$	$[1\ 0\ 0\ 0\ 0\ 0\ 1]$

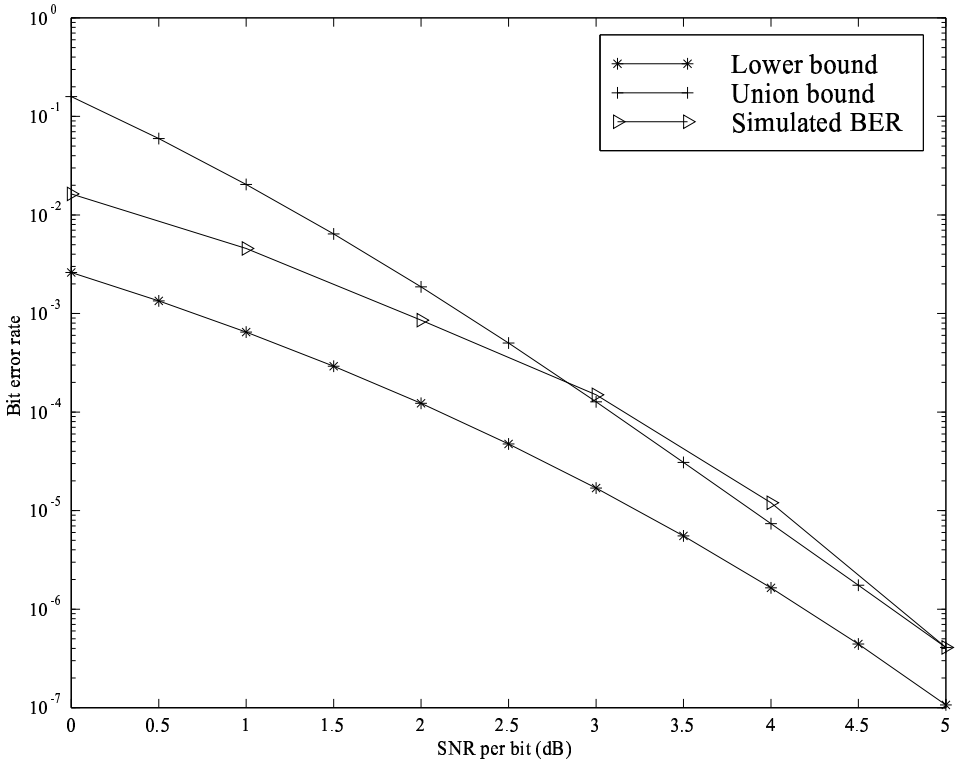

Figure 31: Simulated converged BER of the SCCC shown in Fig. 16 employing a length-36 sub-vector interleaver plotted along with the union bound and a lower bound on the SCCC.

Table 12: Weight-7 interleaver input and output sequences

Interleaver Input $\underline{p}$	Interleaver Output $\underline{p}_\pi$
$\begin{bmatrix}1&1&0\\1&1&0\\0&1&1\\1&0&0\end{bmatrix}, \begin{bmatrix}1&1&1\\0&0&0\\1&0&0\\0&1&0\\1&1&0\end{bmatrix}$	$\begin{bmatrix}1&1&0\\1&0&1\\0&1&0\\1&1&0\end{bmatrix}$
$\begin{bmatrix}1&1&0\\1&1&0\\0&1&1\\1&0&0\end{bmatrix}, \begin{bmatrix}1&1&1\\0&0&0\\1&0&0\\0&1&0\\1&1&0\end{bmatrix}$	$\begin{bmatrix}1&1&1\\0&0&\cdots\\1&0&0\\1&0&\cdots\\0&1&0\\0&1&0\\0&0&\cdots\end{bmatrix}$

Table 13: d_{free} and $d_{free,eff}$ codewords

	Inner Encoder Input	Inner Encoder Output
d_{free}	$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
$d_{free,eff}$	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & \cdots \\ 1 & 0 & 0 \\ 1 & 0 & \cdots \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \cdots \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & \cdots \\ 1 & 1 & 1 \\ 1 & 0 & \cdots \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & \cdots \end{bmatrix}$

in Fig. 32. In Fig. 33, we plotted the ratio of the union bound on the BER for the coding scheme employing the length-24 and length-36 average random interleavers and compared it to the ratio of the union bound on the BER for the coding scheme employing the length-24 and length-36 average sub-vector interleavers. We see in Fig. 33 that the BER decreases approximately 1.43 times faster (roughly equal to the increase in the length of the interleavers) for the average sub-vector interleaver than for the average random interleaver. This is what we described as an increase in the interleaver gain in Section 4.3 due to sub-vector interleaving of the weight-d_{free} and weight-$d_{free} + 1$ codewords from the outer encoder into non-terminating input sequences for the inner encoder.

Table 14: Weight distribution up to weight 48 of SCCC in Fig. 16 employing an average length-24 and an average length-36 sub-vector interleaver. The codeword weight, d, expected input sequence weight, $E[w]$, and expected multiplicity, $E[mult]$, was computed from an ensemble of $100,000$ randomly generated interleavers.

	Length-24		*Length-36*	
d	$E[w]$	$E[mult] \cdot 10000$	$E[w]$	$E[mult] \cdot 10000$
9	3.89	25	3.86	4
10	2.90	9	2.00	1
11	3.85	1044	3.85	195
12	2.94	430	2.91	53
13	3.86	6362	3.96	1445
14	3.27	3682	3.14	601
15	3.88	20696	3.91	5490
16	3.94	20105	3.64	3791
17	4.06	50197	4.00	16143
18	4.59	69212	4.23	16561
19	4.51	102568	4.25	41349
20	5.13	154461	4.77	57131
21	5.17	183954	4.77	107762
22	5.57	238186	5.30	174220
23	5.78	265790	5.41	293380
24	5.97	275248	5.81	490880
25	6.27	274155	6.03	789250
26	6.42	243034	6.34	1271700
27	6.80	198344	6.60	1936800
28	6.93	159948	6.88	2921500
29	7.44	110411	7.16	4163400
30	7.46	72489	7.42	5753100
31	7.85	50095	7.70	7560800
32	9.09	29401	7.96	9489000
33	8.17	14174	8.24	11428000
34	7.79	3643	8.49	12991000
35	8.31	2179	8.76	1417000
36	7.94	131	9.02	14648000
37	–	–	9.27	14374000
38	6	15	9.54	13477000
39	–	–	9.79	11933000
40	–	–	10.05	10064000
41	–	–	10.30	8058700
42	–	–	10.56	6082100
43	–	–	10.81	4366100
44	–	–	11.06	2951000
45	–	–	11.32	1866900
46	–	–	11.56	1118600
47	–	–	11.85	620820
48	–	–	12.19	327670

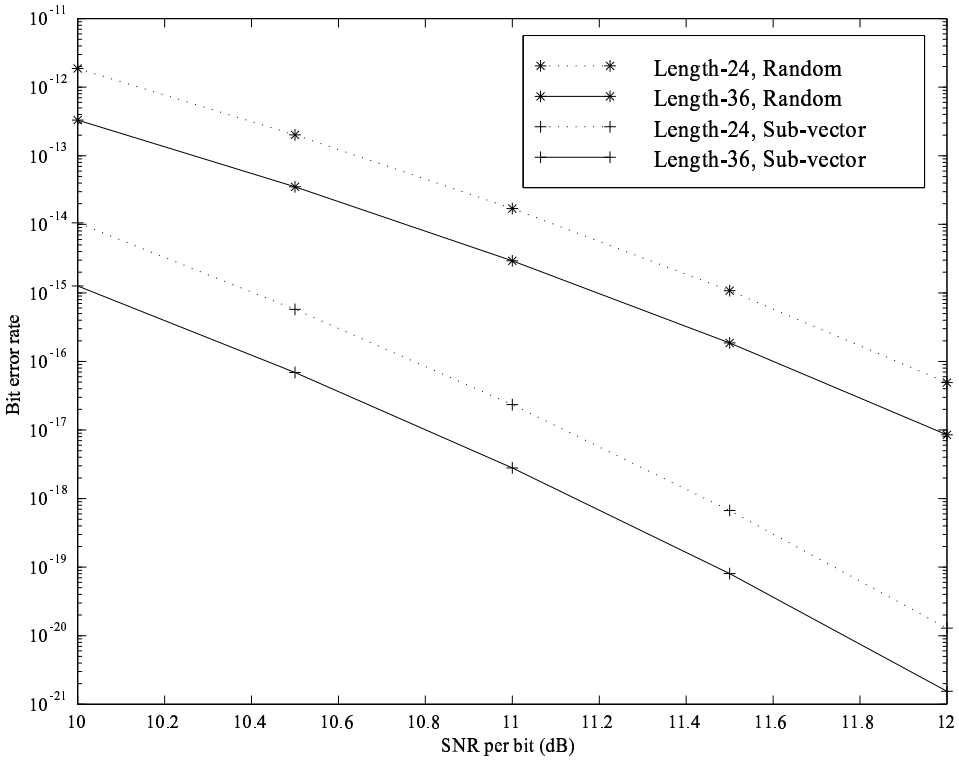

Figure 32: Error floor for the SCCC shown in Fig. 16 employing length-24 and length-36 average random sub-vector interleavers.

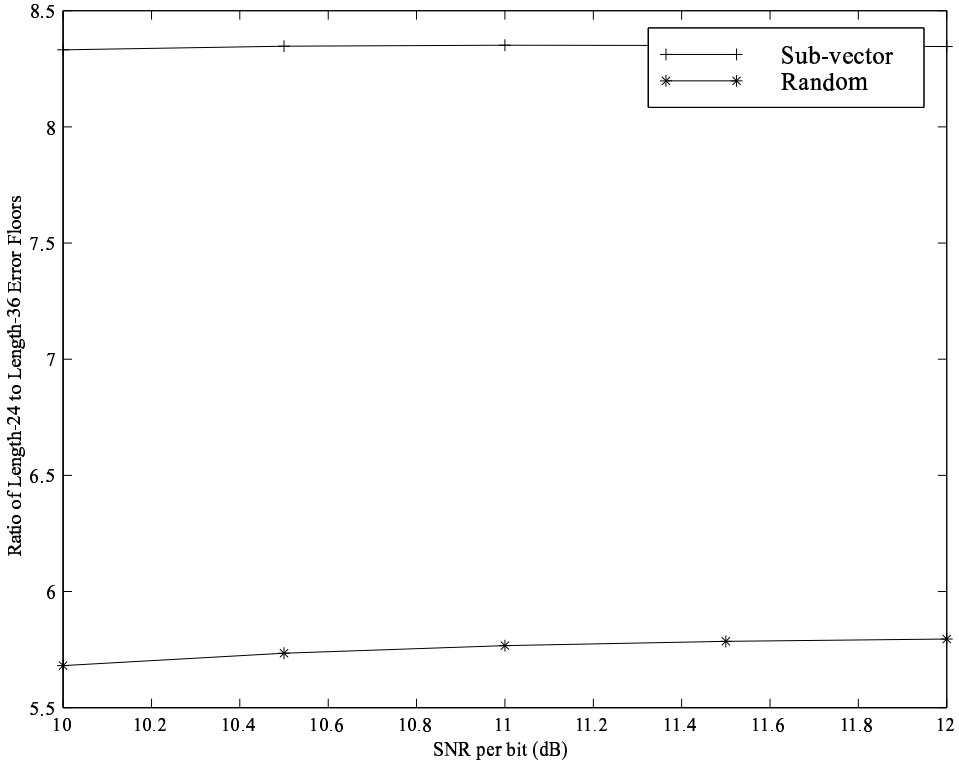

Figure 33: Ratio of length-24 to length-36 error floors for both average random and average sub-vector interleavers for the SCCC shown in Fig. 16.

Table 15: Weight distribution up to weight 48 of SCCC in Fig. 16 employing an average length-24 and an average length-36 random interleaver. The codeword weight, d, expected input sequence weight, $E[w]$, and expected multiplicity, $E[mult]$, was computed from an ensemble of $100,000$ randomly generated interleavers.

	Length-24		*Length-36*	
d	$E[w]$	$E[mult] \cdot 10000$	$E[w]$	$E[mult] \cdot 10000$
7	3.89	41	4.80	11
8	2.90	24	4.48	6
9	3.89	394	4.81	141
10	2.90	614	4.75	169
11	3.85	2563	4.88	913
12	2.94	3717	4.90	1017
13	3.86	9121	4.97	3098
14	3.27	14157	5.07	3928
15	3.88	26566	5.20	8183
16	3.94	44877	5.42	11615
17	4.06	75606	5.63	20879
18	4.59	127113	6.00	33009
19	4.51	197733	6.30	55540
20	5.13	302844	6.79	91721
21	5.17	428350	7.23	151528
22	5.57	581556	7.76	259581
23	5.78	745551	8.23	435621
24	5.97	889280	8.72	745241
25	6.27	1000725	9.19	1250965
26	6.42	1051819	9.62	2083833
27	6.80	1028959	10.04	3389708
28	6.93	939935	10.45	5383614
29	7.44	803775	10.87	8206630
30	7.46	634427	11.27	12095315
31	7.85	467703	11.67	17146812
32	9.09	322422	12.07	23281193
33	8.17	201610	12.46	30250507
34	7.79	116828	12.86	37611086
35	8.31	63358	13.25	44622945
36	7.94	29007	13.64	50564538
37	–	10360	14.03	54667228
38	6	3976	14.42	56360350
39	–	1565	14.80	55395911
40	–	274	15.19	51893789
41	–	–	10.30	46296176
42	–	21	15.57	39339862
43	–	–	15.96	31793380
44	–	–	16.34	24422132
45	–	–	16.73	17800211
46	–	–	17.11	12294438
47	–	–	17.50	8029649
48	–	–	17.88	4940102

CHAPTER VII

CONCLUSION

This chapter is a summary of the contributions of this research. Areas where this research may be continued in future work are also noted here.

7.1 Dynamical System Representation of Turbo Codes

The constituent encoders in a turbo coding scheme can be represented as dynamical systems. In this representation, the input to the dynamical system are the message bits to be encoded, and the states of the dynamical system correspond to the delay states of the encoder. This representation of the overall turbo coding scheme led to new insights in the performance of interleavers in turbo codes. These insights led to a method of interleaving called "sub-vector interleaving."

Future research in this area could include representing the constituent encoders in a turbo-coded modulation scheme as dynamical systems.

7.2 Sub-vector Interleaving

Sub-vector interleaving is a low-complexity method of interleaving that was developed based on the understanding of interleaver performance in turbo codes gained by examining the constituent encoders as dynamical systems. Sub-vector interleavers can be used with both PCCCs and SCCCs. In both of these cases, the free distance codeword can be increased, and the multiplicity of low weight codewords can be decreased. Sub-vector interleavers can be scaled up to any length desired without altering the original design properties and without increasing the complexity of the

method.

Sub-vector interleaving allows for the constituent encoders to be terminated in the zero state by appending a *single* tail sequence to the input sequence. Normally, tail sequences must be applied to the input sequence at each encoder, since interleaving interferes with the terminating tail sequence in most cases.

Distance spectrum analysis of the turbo codes employing sub-vector interleaver showed that sub-vector interleaving increases the average free distance codeword weight and lowers the error floor compared to an average random interleaver. These analytical results were verified by simulation the turbo code performance in a channel corrupted by additive white Gaussian noise.

In the case of SCCCs, the interleaver gain of the coding scheme can be increased with the use of sub-vector interleavers. An upperbound on the amount of increase in the interleaver gain was derived.

Future work in this area could include examining the delay characteristics of a sub-vector interleaver, an issue addressed in [16]. Also, the application of sub-vector interleaving with tail-biting termination to PCCCs or SCCCs could be studied.

Tail-biting termination for a recursive convolutional encoder[1] allows the encoder to be initialized in any state (not just the zero state) as long as the final state of the encoder is the same as the initial state. For example, we can determine the initial and final states of a recursive convolutional encoder that results in tail-biting termination by revisiting the autonomous dynamical system representation of a recursive convolutional encoder in (5):

$$\begin{bmatrix} \underline{v}\,(N) \\ \underline{m}\,(N) \end{bmatrix} = A \cdot \begin{bmatrix} \underline{v}\,(N-1) \\ \underline{m}\,(N-1) \end{bmatrix} = A^N \cdot \begin{bmatrix} \underline{v}\,(0) \\ \underline{m}\,(0) \end{bmatrix}$$

[1]Assume the usual case of a primitive feedback polynomial. The period of a primitive feedback polynomial, σ, is $2^k - 1$, where k corresponds to both the degree of the polynomial and the number of delay states of the encoder.

where

$$\underline{v}\,(0) = \underline{v}\,(N) = \underline{v}, \text{ for tail-biting termination,} \tag{40}$$

$$\underline{m}\,(N) = \left[0\ 0 \cdots 0\right]', \tag{41}$$

and

$$A^N = A^{m\cdot\sigma} = \begin{bmatrix} A_k^N & V_{k,\sigma} & \cdots & V_{k,\sigma} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}.$$

The $k \times k$ matrix, A_k^N, in the left hand corner of A^N is the state transition matrix of the non-autonomous dynamical system representation of the encoder given in (4) to the N^{th} power. Therefore, tail-biting termination is achieved when the summation of the length-σ sub-vectors of the input sequence contained in $\underline{m}\,(0)$, $\underline{m}_\sigma\,(0)$,satisfies the following:

$$\left(I + A_k^N\right)\underline{v} = \left[V_{k,\sigma}\right] \cdot \underline{m}_\sigma\,(0)\,.$$

As usual in the case of PCCCs, error patterns corresponding to codewords generated by terminating input sequences for both encoders are least likely to be corrected by the decoder:

$$\begin{aligned} \underline{v} &= \left(I + A_k^N\right)^{-1}\left[V_{k,\sigma}\right] \cdot \left(\underline{m}_\sigma\,(0) + \underline{e}_\sigma\right) \\ &= \left(I + A_k^N\right)^{-1}\left[V_{k,\sigma}\right] \cdot \underline{m}_\sigma\,(0)\,, \text{ if } \left[V_{k,\sigma}\right] \cdot \underline{e}_\sigma = \underline{0}. \end{aligned}$$

Sub-vector interleavers are designed to permute terminating input sequences to one encoder into non-terminating input sequences to the other encoder. Therefore, sub-vector interleaving should perform well in the case of tail-biting termination just as it did for the case of zero-state termination explored in this research.

7.3 Sub-vector Constrained S-random Interleavers

Incorporating S-random interleaving into the sub-vector interleaving method resulted in another method of interleaving called "sub-vector constrained S-random interleaving." This combination of methods of interleaving caused improvements in the BER performance of turbo codes using either straight S-random or sub-vector interleavers.

The spread parameter, S, is an important design consideration since it governs the amount of spread between bits in the output sequence of the interleaver, but it also affects the time required for the S-random interleaver to generate an interleaver. A maximum value for the S-parameter based on the length of the interleaver was described. The effect of the sub-vector constraint on the spread parameter was also investigated. A technique for randomly generating sub-vector constrained S-random interleavers with maximum spread was presented.

Future work in this area could include investigating prunable sub-vector constrained S-random interleavers, as discussed in [20], for use in applications where various interleaver lengths are required and storage of multiple interleavers is not an option.

7.4 Decoder Convergence

The convergence of the turbo decoding algorithm to maximum-likelihood decisions on the decoded input sequence is required to demonstrate the improvement in BER performance caused by the use of sub-vector interleavers. Convergence to maximum-likelihood decisions by the decoder did not always occur in the regions where it was feasible to generate the statistically significant numbers of error events required to approximate the BER performance for a particular coding scheme employing a sub-vector interleaver. Therefore, a technique for classifying error events by the mode of convergence of the decoder was used to illuminate the effect of the sub-vector interleaver at SNRs where it was possible to simulate the BER performance of the

coding scheme.

Future work in this area could include developing fast simulation methods for turbo coding schemes. Research described in [35] [40] could be adapted for use in turbo coding schemes.

REFERENCES

[1] Agrawal, D. and Vardy, A., "The turbo decoding algorithm and its phase trajectories," *IEEE Trans. Inform. Theory*, vol. 47, pp. 699–722, Feb. 2001.

[2] Andersen, J., "Selection of component codes for turbo coding based on convergence properties," *Annales des Telecommunications*, vol. 54, Mar. 1999.

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

[4] Barbulescu, A. S. and Pietrobon, S. S., "Interleaver design for turbo codes," *Electron. Lett.*, vol. 30, pp. 2107–2108, Dec. 1994.

[5] Barbulescu, A. and Pietrobon, S., "Terminating the trellis of turbo-codes in the same state," *Electronics Letters*, vol. 31, pp. 22–23, Jan. 1995.

[6] Benedetto, S., Divsalar, D., Montorsi, G., and Pollara, F., "A soft-input soft-output app module for iterative decoding of concatenated codes," *IEEE Communications Letters*, vol. 1, pp. 22–24, Jan. 1997.

[7] Benedetto, S., Divsalar, D., Montorsi, G., and Pollara, F., "Serial concatenation of interleaved codes: analysis, design, and iterative decoding," *IEEE Transactions on Information Theory*, vol. 44, pp. 909–926, May 1998.

[8] Benedetto, S. and Montorsi, G., "Serial concatenation of block and convolutional codes," *Electronics Letters*, vol. 32, pp. 887–888, May 1996.

[9] Benedetto, S. and Montorsi, G., "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 409–428, Mar. 1996.

[10] Benedetto, S., Montorsi, G., Divsalar, D., and Pollara, F., "Iterative decoding of serially concatenated codes with interleavers and comparison with turbo codes," in *Proc. 1997 IEEE Global Telecomm. Conf.*, no. 2, pp. 654–658, Nov. 1997.

[11] Berrou, C. and Glavieux, A., "Near optimum error correcting coding and decoding: turbo codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, Oct. 1996.

[12] Berrou, C., Glavieux, A., and Thitimajshima, P., "Near shannon limit error-correcting and decoding: turbo codes," in *Proc.1993 IEEE Int. Conf. Communication*, no. 2, pp. 1064–1070, May 1993.

[13] Breiling, M., Peeters, S., and Huber, J., "Class of double terminating turbo code interleavers," *Electronics Letters*, vol. 35, pp. 389–391, Mar. 1999.

[14] Chi, D., "A new block helical interleaver," in *Proc. 1992 IEEE Military Communications Conf.*, no. 2, pp. 799–804, Nov. 1992.

[15] Daneshgaran, F. and Mondin, M., "Design of interleavers for turbo codes: iterative interleaver growth algorithms of polynomial complexity," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1845–1859, Sept. 1999.

[16] Daneshgaran, F. and Mondin, M., "Optimized turbo codes for delay constrained applications," *IEEE Transactions on Information Theory*, vol. 48, pp. 293–305, Jan. 2002.

[17] Dolinar, S. and Divsalar, D., "Weight distributions for turbo codes using random and nonrandom permutations," *JPL TDA Progr. Rep. 42-122*, pp. 56–65, Aug. 1995.

[18] Dunscome, E. and Piper, F., "Optimal interleaving for convolutional coding," *Electronics Letters*, vol. 25, pp. 1517–1518, Oct. 1989.

[19] Feng, W., Yuan, J., and Vucetic, B., "A code-matched interleaver design for turbo codes," *IEEE Trans. Commun.*, vol. 50, pp. 926–937, June 2002.

[20] Ferrari, M., Scalise, F., and Bellini, S., "Prunable s-random interleavers," in *IEEE Conference on Communications*, vol. 3, pp. 1711–1715, Apr. 2002.

[21] Fragouli, C. and Wesel, R., "Semi-random interleaver design criteria," in *Global Telecommunications Conference*, pp. 2352–2356, 1999.

[22] Garello, R., Pierleoni, P., and Benedetto, S., "Computing the free distasnce of turbo codes and serially concatenated codes with interleavers: Algorithms and applications," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 800–812, May 2001.

[23] Hagenauer, J., Offer, E., and Papke, L., "Iterative decoding of binary block and convolutional codes," *IEEE Transactions on Information Theory*, vol. 42, pp. 429–445, Mar. 1996.

[24] Hagenauer, J., Robertson, P., and Papke, L., "Iterative (turbo) decoding of systematic convolutional codes with the map and sova algorithms," in *Proc. of the 1994 ITG Conference on Source and Channel Coding, Munich*, Oct. 1994.

[25] Hattori, M., Murayama, J., and McEliece, R., "Pseudo-random and self-terminating interleavers for turbo codes," in *Winter 1998 Information Theory Workshop*, 1998.

[26] Herzberg, H., "Multilevel turbo coding with short interleavers," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 303–309, Feb. 1998.

[27] Hokfelt, J., Edfors, O., and Maseng, T., "A survey on trellis termination alternatives for turbo codes," in *1999 IEEE 49th Vehicular Technology Conference*, pp. 2225–2229, July 1999.

[28] Hokfelt, J., Edfors, O., and Maseng, T., "Turbo codes: correlated extrinsic information and its impact on iterative decoding performance," in *1999 IEEE 49th Vehicular Technology Conference*, pp. 1871–1875, July 1999.

[29] Hokfelt, J., Edfors, O., and Maseng, T., "On the theory and performance of trellis termination methods for turbo codes," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 838–847, May 2001.

[30] Khandani, A. K., "Group structure of turbo-codes with application to the interleaver design," in *2000 Canadian Conference on Electrical and Computer Engineering*, pp. 255–259, Mar. 2000.

[31] Khandani, A., "Design of turbo-code interleaver using Hungarian method," *Electronics Letters*, vol. 34, pp. 63–65, Jan. 1998.

[32] List, N. and Williams, D., "Low complexity long interleaver design for turbo codes," in *Allerton Conf. on Communication, Control, and Computing*, Oct. 2001.

[33] List, N. and Williams, D., "Low complexity design of long interleavers for serially concatenated convolutional codes," in *Conf. on Infor. Sciences and Systems*, Mar. 2002.

[34] List, N. and Williams, D., "Improved s-random interleaver design for serially concatenated convolutional codes," in *Canadian Workshop on Infor. Theory*, May 2003.

[35] Orsak, G., "A note on estimating false alarm rates via importance sampling," *IEEE Transactions on Communications*, vol. 41, pp. 1275–1277, Sept. 1993.

[36] Perez, L., Seghers, J., and Costello, D. J., "A distance spectrum interpretation of turbo codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1698–1709, Nov. 1996.

[37] Qi, F., Wang, M., Sheikh, A. H., and Shao, D., "Class of turbo code interleavers based on divisibility," *Electron. Lett.*, vol. 36, pp. 46–47, Jan. 2000.

[38] Reid, A., Gulliver, T., and Taylor, D., "Convergence and errors in turbo-decoding," *IEEE Trans. Commun.*, vol. 49, pp. 2045–2051, Dec. 2001.

[39] Sadjadpour, H., Sloane, N., Salehi, M., and Nebe, G., "Interleaver design for turbo codes," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 831–837, May 2001.

[40] Sadowsky, J., "A new method for viterbi decoding simulation using importance sampling," *IEEE Transactions on Communications*, vol. 38, pp. 1341–1351, Sept. 1990.

[41] Shao, R., Lin, S., and Fossorier, M., "Two simple stopping criteria for turbo decoding," *IEEE Transactions on Communications*, vol. 47, pp. 1117–1120, Aug. 1999.

[42] Sklar, B., "Turbo code concepts made easy, or how I learned to concatenate and reiterate," in *MILCOM 97 Proceedings*, pp. 20–26, Nov. 1997.

[43] Valenti, M., "An introduction to turbo codes," *Published on-line at http://www.csee.wvu.edu/ mvalenti/documents/valenti1996.pdf*, 1996.

[44] Yuan, J., Vucetic, B., and Feng, W., "Combined turbo codes and interleaver design," *IEEE Trans. Commun.*, vol. 47, pp. 484–487, Apr. 1999.