# Maximum Likelihood Decoding Analysis of Accumulate-Repeat-Accumulate Codes

Aliazam Abbasfar[†], Dariush Divsalar[‡] and Kung Yao[†]
[†]Dept. of EE, UCLA
[‡]Jet Propulsion Laboratory[1]

**ABSTRACT**

**Repeat-Accumulate (RA) codes are the simplest turbo-like codes that achieve good performance. However, they cannot compete with Turbo codes or low-density parity-check codes (LDPC) as far as performance is concerned. The Accumulate-Repeat-Accumulate (ARA) codes are obtained by adding a pre-coder in front of RA codes. These codes not only are very simple, but also achieve excellent performance. In this paper, the performance of these codes with (ML) decoding are analyzed and compared to random codes by very tight bounds. Some simple codes are shown that perform very close to Shannon limit with maximum likelihood decoding.**

## 1 Introduction

The advent of Turbo codes; introduced in [1]; has started a big movement towards the invention of a myriad of new code structures. The basic properties of these codes are the ability of iterative decoding and using pseudorandom interleavers. A pretty wide class of code structures called Turbo-like codes was introduced in [2]. Repeat-Accumulate codes (RA) are perhaps the simplest codes among this class. Simplicity of these codes lends itself to a more comprehensive analysis of their performance. Divsalar et al. have shown the performance of these codes with ML decoding and proven that they can achieve near Shannon-limit performance [2]. Moreover, they have proven that it achieves the Shannon-limit when rate goes to zero.

Irregular Repeat-Accumulate (IRA) codes could achieve much better performance, which was shown by Jin [3]. Jin presented a method for designing very good IRA codes for binary erasure and additive white Gaussian channels. He showed that they outperform Turbo codes for codes with very large block sizes.
In this paper first we analyze the performance of RA codes with regular puncturing. We show that with puncturing we can construct better codes as far as the performance is concerned.

Since there is no practical ML decoding algorithm available for block codes with large block size, we use the performance bounds to obtain some insight on codes' behavior. In [4] Divsalar provides a tight upper bound on frame (word) error rate (FER) and bit error rate (BER) for a (n, k) linear block code decoded by Maximum Likelihood criterion over an additive white Gaussian noise (AWGN) channel. It also provides a minimum Eb/No threshold with closed form expression. We use this bound throughout the paper.

## 2 RA codes with puncturing

Repeat-Accumulate codes are the simplest codes among Turbo-like codes, which make them very attractive for analysis. The general block diagram of this code is drawn in Fig. 1. An information block of length N is repeated $q$ times and interleaved to make a block of size qN, and then followed by an accumulator.

Fig. 1: Repeat-Accumulator code block diagram

[1] This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA.

We use the concept of uniform interleaver [5] to compute the overall input-output weight enumerator (IOWE). Therefore, we need to compute the IOWE of both repetition code and the accumulator. For repetition code it is simply the following:

$$A_{w,d}^{rep(q)} = \begin{cases} \binom{N}{w} & ; d = q\,w \\ 0 & ; \text{otherwise} \end{cases}$$

The IOWE of the accumulator is:

$$A_{w,d}^{acc} = \binom{N-d}{\lfloor w/2 \rfloor}\binom{d-1}{\lceil w/2 \rceil - 1}$$

To compute the IOWE of the RA codes with puncturing we use the equivalent encoder depicted in Fig. 2 instead of the accumulator with puncturing. Puncturing uses a periodic pattern 00...0X with period p, where zeros indicate the puncturing positions.

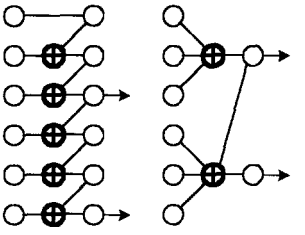

Fig. 2: Accumulator with puncturing and its equivalent graph for p=3.

As we see the equivalent code is a concatenated code of a regular check code and an accumulator, which is shown in Fig. 3.

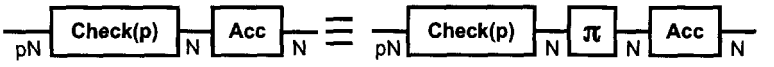


Fig 3. Block diagram of accumulator with puncturing

Since the check code is regular and memoryless, the presence of any interleaver between two codes does not change the IOWE of the overall code. In order to compute the IOWE for this code we insert a uniform interleaver between two codes.

The next step is to compute the IOWE of the check code. The IOWE can be expressed in a simple closed form formula if we use the two dimensional Z-transform. The inverse Z-transform results in $A^c_{w,d}$. We start with N=1, i.e. we have only one parity-check. We have

$$A^c(W,D) = E_p(W) + O_p(W)D$$

Where $E_p = \text{Even}[\,(1+W)^p]$ and $O_p = \text{Odd}[(1+W)^p]$.

Since there are N independent check nodes in the code the IOWE can be written in Z-transform as:

$$A^c(W,D) = (E_p(W) + O_p(W)D)^N = \sum_{d=0}^{N}\binom{N}{d}E_p(W)^{N-d}O_p(W)^d D^d$$

The IOWE is obtained by taking the inverse Z-transform. The closed form expression for Aw,d for arbitrary p is very complicated. Instead we derive the IOWE for p=2,3,and 4, which are practically more useful.

### Case p = 2

Using the general formula in Z-transform we have:

$$A^{c(2)}(W,D) = (1+W^2 + WD)^N$$

It can be expanded as following:

$$\sum_{d=0}^{N}\binom{N}{d}(1+W^2)^{N-d}(2W)^d D^d = \sum_{d=0}^{N}\binom{N}{d}\left[\sum_{j=0}^{N-d}\binom{N-d}{j}W^{2j}(2W)^d\right]D^d$$

Therefore the IOWE can be expressed as

$$A_{w,d}^{c(2)} = \begin{cases} \binom{N}{d}\binom{N-d}{j}2^d & ; w = d+2j \text{ for } j = 0,\ldots,N-d \\ 0 & ; \text{otherwise} \end{cases}$$

### Case p = 3

Starting from general formula in Z-transform we have:

$$A^{c(3)}(W,D) = (1+3W^2 + (3W+W^3)D)^N$$

It can be expanded as following

$$\sum_{d=0}^{N}\binom{N}{d}(1+3W^2)^{N-d}(3W+W^3)^d D^d = \sum_{d=0}^{N}\binom{N}{d}\left[\left(\sum_{i=0}^{N-d}\binom{N-d}{i}3^i W^{2i}\right)\left(\sum_{i=0}^{N-d}\binom{d}{i}3^i (W)^{2(d-i)}\right)W^d\right]D^d$$

Therefore the IOWE is:

$$A_{w,d}^{c(3)} = \begin{cases} \binom{N}{d}\sum_{i=0}^{\min(j,d)}\left[\binom{d}{i}\binom{N-d}{j-i}3^{d+j-2i}\right] & ; w = d+2j \\ 0 & ; \text{otherwise} \end{cases}$$

### Case p = 4

The code for this case can be viewed as a concatenated code as shown in Fig. 4. Because the check code is regular and memoryless, we can put any interleaver between the codes without changing the IOWE of the overall code.

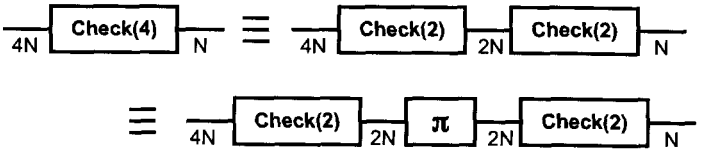


Fig 4. Block diagram of check_4 code and its equivalents

By using a uniform interleaver and the results found for case p=2 the IOWE can be written as:

$$A_{w,d}^{c(4)} = \sum_{h=0}^{2N} \frac{A_{w,h}^{c(2)} A_{h,d}^{c(2)}}{\binom{2N}{h}}$$

This method can be applied for any p that can be decomposed into two smaller numbers.
Having computed the IOWE of the check code, we can use the uniform interleaver formula to compute the IOWE of the accumulator with puncturing. We have

$$A_{w,d}^{acc(p)} = \sum_{h=0}^{N} \frac{A_{w,h}^{c(p)} A_{h,d}^{acc}}{\binom{N}{h}}$$

It should be noted that despite the fact that we use a uniform interleaver to obtain the IOWE, we come up with the exact IOWE for accumulator with puncturing.
The next step is to find the IOWE of the RA code with puncturing, which is derived in case of a uniform interleaver after repetition as:

$$A_{w,d}^{rep(q)\text{-}acc(p)} = \sum_{h=0}^{qN} \frac{A_{w,h}^{rep(q)} A_{h,d}^{acc(p)}}{\binom{qN}{h}}$$

## 2.1 Performance of RA codes with regular puncturing

RA codes are usually non-systematic codes, i.e. the information block is not sent along with the output of the accumulator. However, the RA codes with puncturing should be systematic in order to be decodable by iterative decoding.

Fig. 5 illustrates the normalized distance spectrum of some rate 1/2 codes for a block size of 4000. These codes are RA code (q=2), RA code with puncturing (q=3,p=3), (q=4,p=4), and random code. The thresholds of these codes for infinite block lengths using Divsalar's bound are compared in Table I. Discrepancy between Random code threshold and Shannon limit is due to the upper bound which is slightly loose for rate 1/2.

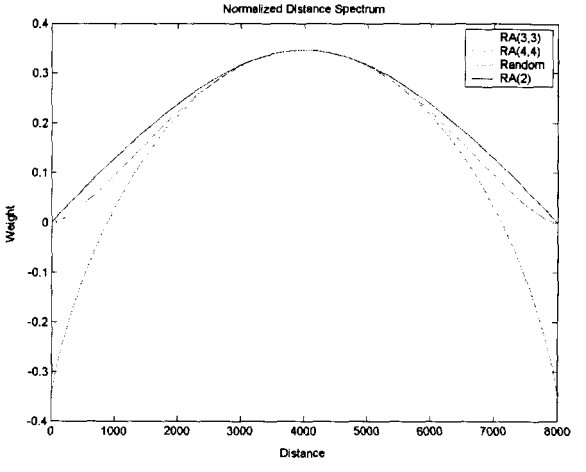


Fig. 5: Normalized distance spectrum of RA codes with puncturing

Table I

| Eb/No threshold | RA(q=2) | RA_punc(q=3,p=3) | RA_punc(q=4,p=4) | Random Code | Shannon limit |
|---|---|---|---|---|---|
| Rate 1/2 | 3.38 dB | 1.46 dB | 0.85 dB | 0.297 dB | 0.187 dB |

# 3 ARA codes

In this section we investigate the effect of an accumulator precoder on the performance of RA codes. It is shown that we can obtain some ARA codes (as precoded RA) that achieve extremely near Shannon capacity performance with ML decoding. Unfortunately, the iterative decoding algorithm cannot achieve the same performance. However, there are still some good codes that have performance comparable to turbo codes. The main advantages of these codes are the simplicity and the capability of parallelization. Therefore, practical high speed decoders can be conceived for these codes.

The accumulator precoder is a rate one encoder used to improve the performance of the RA codes. However, only a portion of the information block goes to the accumulator. In other words, M bits are passed through without any change and the rest (N-M bits) goes through an accumulator. Then the overall output bits are applied to the punctured RA code. This is essential for iterative decoding in order to start the message-passing algorithm. M is considered a parameter in code design. The effect of this parameter is studied in ML decoding. Fig. 6 (a) shows a block diagram of the precoder. Fig. 6 (b) shows the ARA code.

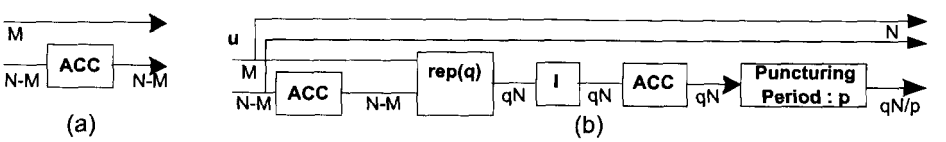


Fig. 6(a): The block diagram of the precoder (b) ARA code

In order to find the performance of the code we need to compute the IOWE of the precoder. It is easily computed using the IOWE of the accumulator code as follows:

$$A^{pre}_{w,d} = \sum_{m=0}^{M} \binom{M}{m} A^{acc}_{w-m,d-m}$$

Therefore the IOWE of the overall code can be written as:

$$A^{pre\text{-}rep(q)\text{-}acc(p)}_{w,d} = \sum_{h=0}^{N} \frac{A^{pre}_{w,h} A^{rep(q)\text{-}acc(p)}_{h,d}}{\binom{N}{h}}$$

## 3.1 Performance of ARA codes

The Divsalar BER performance bound for ARA(3,3) and ARA(4,4) for different Ms are compared to that of random code for the same block size (4000) in Fig. 8. It is observed that the more number of bits accumulates in the precoder, the lower the code threshold becomes. However, the improvement stops at certain point, which is M= 1/5 N for ARA(3,3) and M= 2/5 N for ARA(4,4). It is obvious that when M=N the codes turn into RA with puncturing. It is very interesting that the performance of the ARA(4,4) approaches very closely to that of random codes for the same block size in low Eb/No region.

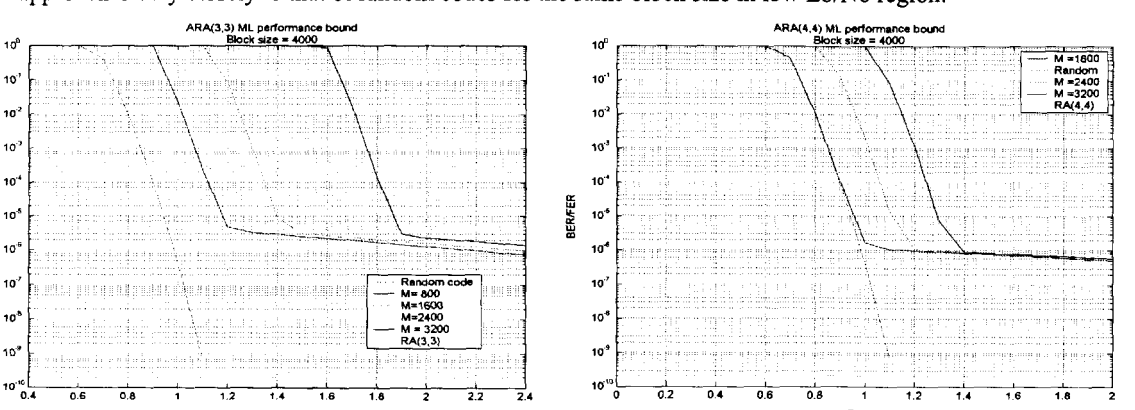


Fig. 7: BER bounds for ARA(3,3) and ARA(4,4) codes

It is very instructive to observe the distance spectrum of these codes (For optimum M). As we see in Fig. 8 the only difference between the distance spectrum of these codes and a random code is in the low-distance region, which causes the error floor.

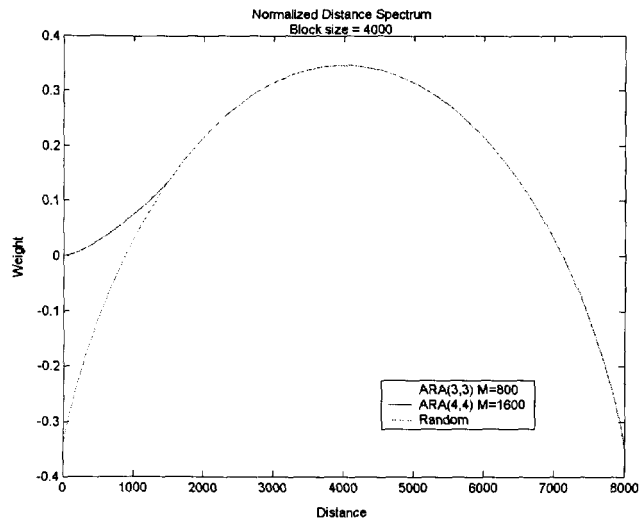


Fig. 8: Normalized distance spectrum of ARA codes with puncturing

Table II tabulates the Divsalar's min. Eb/No threshold for the same codes discussed. As we expected based on the BER performance bound for input block of 4000 that was shown, the threshold of ARA(4,4) for infinite block is also extremely close to the threshold of random codes. Discrepancy between Random code threshold and Shannon limit is due to the upper bound which is slightly loose for rate 1/2.

Table II

| Eb/No threshold | ARA_punc(q=3,p=3) | ARA_punc(q=4,p=4) | Random Code | Shannon |
|---|---|---|---|---|
| Rate 1/2 | 0.509 dB | 0.299 dB | 0.297 dB | 0.187 dB |

# 4 References


[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: turbo codes," *Proc. 1993 IEEE International Conf. On Communications*, Geneva, Switzerland, May 1993, pp. 1064-1070.

[2] D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for Turbo-like Codes," *Proc. 36th Allerton Conf. On Communication, Control and Computing*, Sept. 1998, pp. 201-210.

[3] H. Jin, Analysis and Design of Turbo-like Codes. Ph.D. Thesis, California Institute of Technology, Pasadena, 2001.

[4] D. Divsalar, "A Simple Tight Bound on Error Probability of Block Codes with Application to Turbo Codes," *JPL TMO Progress Report 42-139*, Nov. 1999, pp.1-35. (http://tmo.jpl.nasa.gov/tmo/progress_report/42-139/title.htm)

[5] S. Bennedetto and G. Montorsi, "Unveiling Turbo Codes: Some results on parallel concatenated codes," *IEEE Trans. On Information Theory*, vol. 42, no. 2, March 1996, pp. 409-428.