Nonsystematic Convolutional Codes for Sequential Decoding In Space Applications

Kwame Ackah Bohulu

November 14, 2019

1 Introduction

1. About the Deep-Space Channel

   - Channel Model: Additive White Gaussian Noise Channel
   - Decoder for Convolutional Codesl: Sequential Decoder(Best at that time)

2. Purpose of Paper

   - Introduce New class of Convolutional codes
   - Has Lower BER and lower implementation complexity than what is used currently

3. Terms to be used in the Paper

   - $\mathbf{x} = \{x_0, x_1, x_2, ...\}$: Binary information bit sequence. In polynomial form (D transform) it may be written as $\mathbf{X}(D) = x_0 + x_1 D + x_2 D^2 + ...$
   - rate ($\mathbf{R} = 1/n$) convolutional encoder with memory order m: convolutional encoder with single input and n outputs.
   - The n outputs are determined by the n transfer functions (also known as the generator functions) . The "i"th transfer function is written as $\mathbf{G}^{(i)}(D) = g_0^i + g_1^i D + ... + g_m^i D^m$
   - The "i"th input bit is is used to produce the "j"th encoded sequence $\mathbf{C}^{(j)}(D) = c_0^i + c_1^i D + c_2^i D^2 + ...$
   - in compact form

     $$\mathbf{C}^{(j)}(D) = \mathbf{X}(D)\mathbf{G}^{(i)}(D) \tag{1}$$

     All additions and multiplications are done in modulo 2.
   - A convolutional code is said to be systematic if

     $$\mathbf{C}^{(1)}(D) = \mathbf{X}(D) \text{ or } \mathbf{G}^{(1)}(D) = 1 \tag{2}$$

   - initial code word[2] : span of $(m+1)$n encoded digits generated by the encoder with respect to the first information bit x_0
   - minimum distance (d_m): Fewest number of positions in which two initial code words arising from different values of i_0 are different. This is equivalent to the fewest number of nonzero digits in any initial code word with $i_0 = 1$

2 Systematic Versus Non-Systematic Codes

Let the "j"th hard decisioned signal at the receiver be expressed as $\mathbf{R}^{(j)}(D) = r_0^i + r_1^i D + r_2^i D^2 + ...$ which may be rewritten as

$$\mathbf{R}^{(j)}(D) = \mathbf{C}^{(j)}(D) + \mathbf{E}^{(j)}(D) \tag{3}$$

where $\mathbf{E}^{(j)}(D) = e_0^i + e_1^i D + e_2^i D^2 + ...$ is the sequence of errors in the hard decisions.

1. Why systematic codes over non systematic codes?

    - Easy to get estimates of information sequence for synchronization purposes without decoding, ie $\mathbf{R}^{(1)}(D) = \mathbf{X}(D) + \mathbf{E}^{(1)}(D)$
    - According to [3], for any non-systematic codeword, there exists a systematic code word with the same minimum distance d_m. Therefore, there is no extra advantage when using threshold decoders. Threshold decoders make decisions on i_0 based on the initial codeword only[2].
    - There is a fear that non-systematic encoders are more difficult to implement and that catastrophic errors may occur from the recovery of the data.

2. Why non-Systematic codes?

    - According to results from[4]-[6] they have a naturally Lower error probability than that of systematic codes when used with sequential decoding [7],[8] and maximum likelihood decoders[9].
    - The error probalility for such decoders is closely related to the free distance of the code which is equivalent to the fewest number of non-zero elements in a encoded sequence.
    - According to [5], for a given memory order m, non-systematic codes provide more free distance than their systematic equivalents

Based on these considerations, the author seeks to design non-systematic convolutional codes which has good performance with sequential decoding and has the ability to extract the information sequence with as much ease as possible.

3 Quick-Look-In Non-systematic Codes

1. According to [10], a convolutional encoder with a feedfoward (FF) inverse is a necessary and sufficient condition to avoid catastrophic error propagation.

2. FF inverse for $\mathbf{R} = 1/n$ encoder : n-input single output linear sequential circuit with polynomial transfer function $\mathbf{P}_j(D), j = 1, 2, ..., n$ such that

$$\sum_{j=1}^{n} \mathbf{P}_i(D)\mathbf{C}^{(j)}(D) = D^L\mathbf{X}(D) \tag{4}$$

3. since $\mathbf{C}^{(j)}(D) = \mathbf{X}(D)\mathbf{G}^{(i)}(D)$, the above equation may be rewritten as

$$\sum_{i=1}^{n} \mathbf{P}_i(D)\mathbf{G}^{(i)}(D) = D^L \tag{5}$$

4. Case where R

   - We may rewrite eqn (4) and (5) as

   $$\mathbf{P}_1(D)\mathbf{C}^{(1)}(D) + \mathbf{P}_2(D)\mathbf{C}^{(2)}(D) = D^L\mathbf{X}(D) \tag{6}$$

   and

   $$\mathbf{P}_1(D)\mathbf{G}^{(1)}(D) + \mathbf{P}_2(D)\mathbf{G}^{(2)}(D) = D^L \tag{7}$$

   - estimate of $\mathbf{X}(D)$ can be obtained by passing the hard decisioned received signal through the FF inverse, ie

   $$\mathbf{P}_1(D)\mathbf{R}^{(1)}(D) + \mathbf{P}_2(D)\mathbf{R}^{(2)}(D) = D^L[\mathbf{X}(D) + \boldsymbol{\Delta}(D)] \tag{8}$$

   where

   $$\boldsymbol{\Delta}(D) = \delta_0 + \delta_1 D + \delta_2 D^2 + ...$$

   is the sequence of errors in the estimated information digits.

   - using the fact that $\mathbf{R}^{(j)}(D) = \mathbf{C}^{(j)}(D) + \mathbf{E}^{(j)}(D)$, we obtain the equation below.

   $$\mathbf{P}_1(D)\mathbf{E}^{(1)}(D) + \mathbf{P}_2(D)\mathbf{E}^{(2)}(D) = D^L\boldsymbol{\Delta}(D) \tag{9}$$

   This equation relates the errors in the hard decisioned received signal to the errors in the estimated information digits.

   - If we assume that the individual hard decisioned received bits have a probability p, it follows from (9) that if $p << 1$ then $\delta_i = 1$ with probability

   $$p_\delta = \{W[\mathbf{P}_1(D)] + W[\mathbf{P}_2(D)]\}p \tag{10}$$

   where $W[\mathbf{P}_j(D)]$ is the hamming weight of the $[\mathbf{P}_j(D)]$. we refer to

   $$A = \{W[\mathbf{P}_1(D)] + W[\mathbf{P}_2(D)]\}$$

   as the error amplification factor

5. For systematic codes A takes on the the minimum possible value of 1, where $\mathbf{P}_1(D) = 1$ and where $\mathbf{P}_2(D) = 0$

6. For non systematic codes the minimum possible value for A is 2, where $\mathbf{P}_1(D) = \mathbf{P}_2(D) = 1$

7. For such a code, we may rewrite (8) as

$$\mathbf{R}^{(1)}(D) + \mathbf{R}^{(2)}(D) = D^L[\mathbf{X}(D) + \boldsymbol{\Delta}(D)] \tag{11}$$

   which means that the estimate of the information sequence is obtained by the modulo 2 addition of the received sequences. It is worth noting that $\mathbf{P}_1(D) = \mathbf{P}_2(D) = 1$ is only possible if

$$\mathbf{G}^{(1)}(D) + \mathbf{G}^{(2)}(D) = D^L \tag{12}$$

   and any rate $\mathbf{R} = 1/n$ non-systematic code which satisfies this condition is known as a quick-look-in code.

4 Sequential Decoding and Consideration And Code Construction

When it comes to sequential decoding of convilutional codes, there are two important factors to be considered. The distribution of computation and the undetected error probability.

1. Distribution of computation : This value happens to be a random number and the code parameter which is closely related to this is know as the column distance $d_k,\ k = 0, 1, ...$

   - By definition[11], the column distance is minimum distance of the code of memory order k obtained by dropping all terms of degree greater than k from the original code generating polynomial.
   - Examples : For a rate R convolutional encoder, assume we have $\mathbf{G}^{(1)}(D) = g_0^{(1)} + g_1^{(1)}D + g_2^{(1)}D^2$ and $\mathbf{G}^{(2)}(D) = g_0^{(2)} + g_1^{(2)}D + g_2^{(2)}D^2$.
   - We see that $d_0 = 2$ iff $g_0^{(1)} = g_0^{(2)}$ and $d_1 = 3$ iff $d_0 = 2$ and $g_1^{(1)} \neq g_1^{(2)}$ According to [8] simulations have shown that for $d_1 < 3$ the amount of computation required is unaceptablly large and thus in the construction of the quick-look-in codes, the search is restricted to $d_1 > 3$. Specifically

$$g_0^{(1)} + g_1^{(1)} = 1$$

     and

$$g_0^{(2)} + g_1^{(2)} = 1 + D$$

- furthermore, simulations have shown that for good computational performance $d_2, d_3, ...$ should grow as rapidly as possible to minimize the need for long searches in the code trellis. This means the values for $g_2^{(1)}, g_3^{(1)}, ...g_m^{(1)}$ and $g_2^{(2)}, g_3^{(2)}, ...g_m^{(2)}$ should be chosen to maximize the values of the higher column distances.
- Unfortunately, this does not uniquely define the code since many different combinations can lead to the same value of d_k

2. Undetected error probability

   - This parameter is most affected by the free distance of the code which is in turn related to a large number of "1"'s in the generator polynomials.
   - This suggest that for the same value of d_k, you should choose the one which has the larger number of "1"'s in the generator polynomial.

3. with the above considerations in made, a unique QLI code can be constructed using the algorithm below

   (a) choose $g_0^{(1)} = g_0^{(2)} = 1$ and $g_1^{(1)} = 0$. Set $d_1 = 3, k = 2$

   (b) set $g_k^{(1)} = g_k^{(2)} = 0$ and compute d_k. if $d_k > d_{k-1}$ go to step (d).

   (c) set $g_k^{(1)} = g_k^{(2)} = 1$

   (d) if $k = m$, stop. Otherwise $k = k + 1$ and go to step (b)

4. Using the above algorithm, the code with memory order $m = 35$ has the following generator polynomials written in octal notation

$$\mathbf{G}^{(1)} = [533, 533, 678, 737]_8$$

and

$$\mathbf{G}^{(2)} = [733, 533, 678, 737]_8$$

5 Performance with Sequential Decoding

The QLI code for $m = 35$ obtained using the previously stated algorithm is compared with the best known $m = 35$ systematic code which has genetator polynomials

$$\mathbf{G}^{(1)} = [400, 000, 000, 000]_8$$

and

$$\mathbf{G}^{(2)} = [715, 473, 701, 317]_8$$

1. Simulation parameters:

   - Channel models: AWGN and Binary Symetric channel

- Frame Length :256, with 35 extra zero bits to force encoder back to zero state.
- Decoding algorithm: Fano sequential algorithm with maximum of $50,000$ computations per frame. $1,000$ frames decoded for each channel model

2. In conclusion, the QLI codess give comparable computational performance to that of the systematic code with the same memory order, but have a lower undetected error probability.

6 Encoder Implementation

1. To implement any convolutional encoder requires $W[\mathbf{G}^{(1)}(D)]+W[\mathbf{G}^{(2)}(D)]-2$ two-input modulo 2 adders. This is shown in fig. 1
2. This means the QLI requres 53 modulo 2 adders whiles the systematic code requires 21.
3. a simple trick can be used to deal with the large number of modulo 2 adders required. All we need to do is to implement the binary compliment of the desired encoder and and add a circuit that does the necessary complementation at the output.
4. Let $\bar{\mathbf{G}}^{(1)}(D) = \bar{g}_0 + \bar{g}_1 D + \bar{g}_2 D^2 + ... + \bar{g}_m D^m$, where $\bar{g}_i = g_1 + 1$, we may write

   $$\mathbf{G}(D) = \bar{\mathbf{G}}^{(1)}(D) + 1 + D + ... + D^m$$

   or

   $$\mathbf{G}(D) = \bar{\mathbf{G}}^{(1)}(D) + (1 + D^{m+1})/(1 + D)$$

5. This means that the required number of modulo 2 adders is now given by $W[\bar{\mathbf{G}}(D)] + 2 = m + 2 - W[\mathbf{G}(D)]$ and the circuit looks that in fig. 2

   For the QLI code used in the simulation, the number of modulo 2 adders to implement $\mathbf{G}^{(2)}(D)$ and $\mathbf{G}^{(1)}(D)$ reduces to 10 and 11 respectively.

6. In conclusion, the circuit required for the implementation of the QLI isn't too complicated when compared to that of the systematic code with the same memory order.

7 Conclusions

1. The reason for the design of QLI was explained and an algorithm for the construction of such codes was introduced for $R = 1/2$.

2. it was shown that these codes have a lower undetected error probability for sequential decoding when compared to systematic codes of the same order.

3. It was shown that the encoder construction was simpler that that of the systematic encoder construction