专 题 汇 编

ICC' 调制与编码

北京邮电学院图书馆

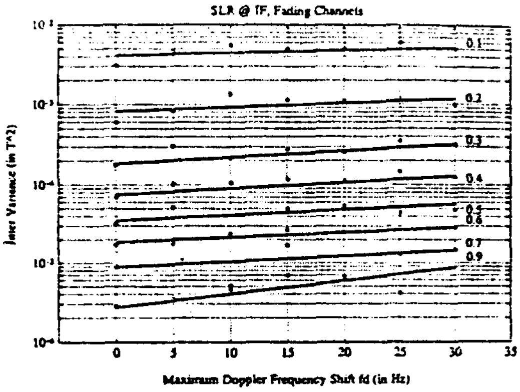


Fig. 6 Jitter performance with SLR in Rayleigh fading channels.

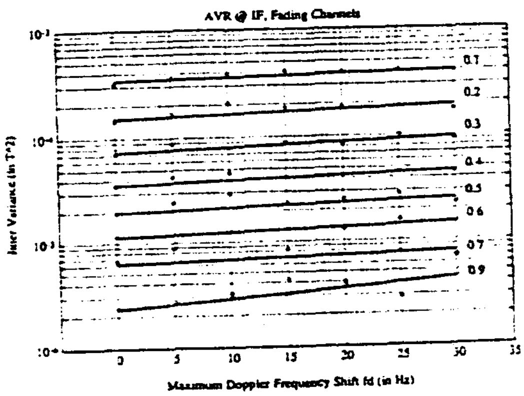


Fig. 7 Jitter performance with AVR in Rayleigh fading channels.

BOUNDS FOR CYLINDRICAL PROJECTION CODES

Gang Yang*, Dong Li and Donald L. Schilling

Department of Electrical Engineering
The City College of New York
New York, NY 10031, USA

ABSTRACT Some methods of analyzing the error performance of cylindrical Projection codes are put forward. The main features of these methods are

1) To use combinatorial generating functions to obtain the weight distribution of a cylindrical Projection code.

2) To make use of the weight distribution of a cylindrical Projection code to obtain the upper bound and lower bound of the code.

It is suggested that the same procedure could also be used to obtain the error performance of an ordinary block Projection code with only a little change of the computer program.

I. INTRODUCTION

Application of error-correcting codes in communication systems is now widely done in modern communication to obtain high performance with small transmitting power. There are many different error-correcting codes. It is well known that the more parity bits, the more errors can be corrected. However, the number of parity bits cannot be arbitrarily large, because the more parity bits per codeword, the less energy per bit, and therefore, the higher the input bit error rate. So, looking for a code which is closer to Shannon's limit is one of the major direction of modern communication research.

The Projection code is an attempt to obtain an ideal code. The Projection code is very easy to encode and decode, and from simulations, it has been determined to have very good performance. All of these reasons make it a desirable code. However, because of its very abrupt performance curve, it is difficult to investigate and evaluate its performance at reasonable input error rates. Therefore, it is necessary to analyze this code theoretically.

The Projection code has three basic forms [1][2]. The data rows are put on the top of the parity rows and the parity checks are made along the slope lines. The Projection Code can be a convolution-like code or a block-like code. For block Projection code, there can be either a cylindrical block code, where the number of the bits in a block is finite, but with the end and the head connected together, or a noncylindrical block code, that is, a block code with a beginning and an end. In this paper, only the cylindrical block Projection code is considered.

The first form is called the Basic Projection Code, or P1 code. In such a block code, the parity bits check only the information bits on the same parity line and do not check any other parity bits.

The second form is called the Partial Autoconcatenation Projection Code, or P2 code. In this code, the parity bits on a given row not only check the information bits, but also check the other parity bits which are on the rows closer to the data and on the same parity check line.

The third form is the Total Autoconcatenation Projection Code, or P3 code. In the case of P3, the parity bits check not only the information bits, the parity bits on the rows above them, but also the parity bits on rows below them.

In this paper, combinatorial technology has been used to obtain the weight distribution of the Projection code. And then, from the weight distribution of that code, the upper bound and lower bound of that code has been obtained.

II. THE METHODS FOR UPPERBOUNDING AND LOWERBOUNDING THE PROJECTION CODE

If the weight distribution of a code is given, the output bit error rate bounds of the optimum decoder can be obtained [3]. Denote M_1 as the number of the codewords with weight w_1, M_2 as the number of the codewords with weight w_2, ..., M_m as the number of codewords with weight w_m, where $w_1 < w_2 < w_3 ... < w_m$ and m is the maximum weight of the code. Since the Projection code is a linear code, it is necessary to consider transmitting the codeword with all zero elements.

Let C(j) stand for the j^{th} codeword, and $|C(j)|$ stand for its weight. Let V be the received codeword, p_b be the input bit error rate and P_b be the output bit error rate. Let C_w, stand for the codeword with weight w, and M stand for the total number of codewords.

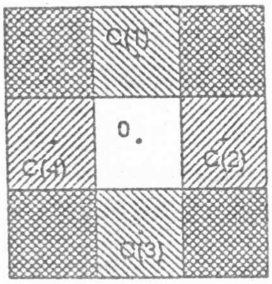
A. THE DERIVATION OF THE UPPERBOUND

Suppose the codeword with all zero elements is transmitted. When the received word is closer to C(j) than to any other codeword, the optimum decoder will think C(j) is transmitted. Hence, $|C(j)|$ errors will be produced. If the received word is at the same distance from two codewords, $|C(j)|$ and $|C(l)|$, and is further from any other codeword, then the optimum decoder will think $|C(j)|$ is transmitted with

---

This research represents a portion of the dissertation submitted to the faculty of the School of Engineering of CUNY in partial fulfillment of the requirements of the Ph.D in EE.

the probability 50% and $|C(l)|$ is transmitted with the probability 50%. So $|C(j)|$ errors and $|C(l)|$ errors will be produced with the probability 50% respectively.



The Weight of C(1),C(2),C(3) and C(4) is W

Fig.1 The Diagram for Upper Bound Analysis

Suppose there are four codewords with the minimum Hamming weight and the transmitted codeword is the all zero codeword which is called codeword 0. This is illustrated in Fig.1. If the received code V is closer to some C(j) than to codeword 0, the optimum receiver will make a wrong decision. The probability that the received code is closer to C(j) than to codeword 0 is the probability of the received code V falling into the half plane cut by the bisect line of C(j) and 0 and on the side of C(j). The shaded area is the area where a wrong decision will be made if the received code V falls into the area. N times the output bit error probability is equal to the summation of the product of the probability of V falling into every rectangle and the weight of the codeword in that rectangle. The probability that V falls into the rectangle of C(j) is less than the probability that V is closer to C(j) than to 0. Based on this discussion, an upper bound can be found.

Mathematically, it can be shown that

$$NP_b=\sum_{j=1}^{M}|C(j)|\cdot P\{|C(j)-V|<|C(i)-V|,\forall i\neq j$$

/transmitting all zero codeword}

$$+\sum_{j=1}^{M}\frac{1}{2}|C(j)|\cdot P\{|C(j)-V|=|C(i)-V|,\text{if such } i \text{ exist}$$

$$\text{and } |C(j)-V|<|C(l)-V| \text{ for} \forall l\neq i \text{ and } \forall l\neq j$$

/transmitting all zero codeword}

$$\leq\sum_{j=1}^{M}|C(j)|\cdot P\{|C(j)-V|\leq|V| \qquad (1)$$

/transmitting all zero codeword}

$$+\frac{1}{2}\sum_{j=1}^{M}|C(j)|\cdot P\{|C_j-V|=|V|$$

/transmitting all zero codeword}

$$=\sum_{j=1}^{m}N_j\cdot w_j\cdot P\{|C_{w_j}-V|<|V|$$

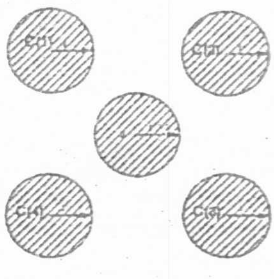
/transmitting all zero codeword}

$$+\frac{1}{2}\sum_{j=1}^{m}N_j\cdot w_j\cdot P\{|C_{w_j}-V|=|V|$$

/transmitting all zero codeword}

B. THE DERIVATION OF THE LOWERBOUND

Let us draw a circle with radius t around every codeword as shown in Fig 2. Any received code falling into the circle with radius t of a codeword, will be corrected to this codeword. If the weight of the code is w, then w errors will be made. Further, suppose there are no errors outside the shaded area. Obviously, $P_b>\frac{1}{N}\sum_{w=0}^{m}|C_w|wP_A(w)$, where $P_A(w)$ stands for the probability of the received word falling into the circle of a codeword with weight w.



The Weight of C(1),C(2),C(3) and C(4) is W

Fig.2 The Diagram for the Lower Bound Analysis

Let $N(w,l;s)$ be the number of error patterns of weight l that are at distance s from a codeword of weight w and d_{min} be the minimum weight of the codewords. For linear codes, $N(w,l;s)$ is the same for every codeword of weight w. It can be shown that in the binary case, [4]

$$N(w,l;s)=\sum_{\substack{j=s-k\geq 0\\ k=l+j-w\geq 0}}\binom{N-w}{k}\binom{w}{j} \qquad (2)$$

and therefore,

$$P_A>\sum_{s=0}^{d_{min}-1}\sum_{l=0}^{N}p_b^l(1-p_b)^{N-l}N(w,l;s)$$

$$+\frac{1}{2}\sum_{l=0}^{N}p_b^l(1-p_b)^{N-l}N(w,l;d_{min}) \qquad (3)$$

$$P_b>\frac{1}{N}\sum_{w=0}^{m}\sum_{l=0}^{N}\sum_{s=0}^{d_{min}-1}w|C_w|p_b^l(1-p_b)^{N-l}N(w,l;s)$$

$$+\frac{1}{2N}\sum_{w=0}^{m}\sum_{l=0}^{N}d_{min}|C_{d_{min}}|p_b^l(1-p_b)^{N-l}N(w,l;d_{min})$$

III. APPLICATION GENERATING FUNCTION TO PROJECTION CODES

It is known that generating functions are a convenient tool for handling selection and arrangement problems with some constraints [5],[6]. The Projection code has a high degree of structure and seeking codewords and pseudocodewords which satisfy the parity check equations is a type of selection. Therefore, it is a useful trial to apply combinatorial procedures to obtain the weight distribution of the extended code.

Let us start with an example. In this example, we will find the number of solutions with various weights of following equations

$$\begin{aligned} &x_1 + x_2 + x_3 + x_4 = 0 \bmod 2 \\ &x_2 + x_3 + x_4 + x_5 = 0 \bmod 2 \\ &x_3 + x_4 + x_5 + x_1 = 0 \bmod 2 \\ &0 \le x_1 \le 1 \\ &0 \le x_2 \le 1 \\ &0 \le x_3 \le 1 \\ &0 \le x_4 \le 1 \\ &0 \le x_5 \le 1 \end{aligned} \tag{4}$$

There are three equations. Each equation will affect the solution and therefore they must be considered together. Let us use x_i^j to stand for $x_i = j$, w_i^j to stand for the summation of weight in the ith equation to equal j, and use w^j to stand for the summation of all variables to equal j.

In this example, the variables are binary, therefore j has only two possible values, namely 0 or 1. We can use $x_1^0 + x_1^1$ to stand for the generating factor of x_1 and let the weight be equal to the exponent of the variable. But in order to see the weight directly, we would like to use the exponents of w_i to stand for the summation of all the variables of ith equation. Therefore, use $x_1^0 + x_1^1 w_3^1$ to stand for the generating factor of x_1 in the third equation, and so on.

The operational rules for the generating function are then

$$x_i^j \cdot x_k^l = \begin{cases} x_i^j w^{-l}, & i = k \text{ and } j = l \\ x_i^j \cdot x_k^l, & i \ne k \\ 0, & i = k,\ j \ne l \end{cases}$$

$$0 \cdot x_i^j = 0 \tag{5}$$

$$w^j \cdot w^l = w^{j+l}$$

$$w_i^j \cdot w_k^l = \begin{cases} w_i^{j+l}, & i = k \\ w_i^j \cdot w_k^l, & i \ne k \end{cases}$$

The rules are based on the following:
1). The weight of any variable can be counted only once;
2). Any variable can take only one value at a time;
3). The weight of the summation of two variables is equal to the summation of the weight of each variable.

For the first equation the generating function is

$$\begin{aligned} g_1 &= (x_1^0 + x_1^1 w_1)(x_2^0 + x_2^1 w_1)(x_3^0 + x_3^1 w_1)(x_4^0 + x_4^1 w_1) \\ &= x_1^0 x_2^0 x_3^0 x_4^0 + x_1 x_2 x_3^0 x_4^0 w_1^2 + x_1 x_2^0 x_3 x_4^0 w_1^2 \\ &\quad + x_1 x_2^0 x_3^0 x_4 w_1^2 + x_1^0 x_2 x_3 x_4^0 w_1^2 + x_1^0 x_2 x_3^0 x_4 w_1^2 \\ &\quad + x_1^0 x_2^0 x_3 x_4 w_1^2 + x_1 x_2 x_3 x_4 w_1^4 \end{aligned} \tag{6}$$

For the second equation the generating function is

$$\begin{aligned} g_2 &= x_2^0 x_3^0 x_4^0 x_5^0 + x_2 x_3 x_4^0 x_5^0 w_2^2 + x_2 x_3^0 x_4 x_5^0 w_2^2 \\ &\quad + x_2 x_3^0 x_4^0 x_5 w_2^2 + x_2^0 x_3 x_4 x_5^0 w_2^2 + x_2^0 x_3 x_4^0 x_5 w_2^2 \\ &\quad + x_2^0 x_3^0 x_4 x_5 w_2^2 + x_2 x_3 x_4 x_5 w_2^4 \end{aligned} \tag{7}$$

The generating function for the third equation is

$$\begin{aligned} g_3 &= x_1^0 x_3^0 x_4^0 x_5^0 + x_1 x_3 x_4^0 x_5^0 w_3^2 + x_1 x_3^0 x_4 x_5^0 w_3^2 \\ &\quad + x_1 x_3^0 x_4^0 x_5 w_3^2 + x_1^0 x_3 x_4 x_5^0 w_3^2 + x_1^0 x_3 x_4^0 x_5 w_3^2 \\ &\quad + x_1^0 x_3^0 x_4 x_5 w_3^2 + x_1 x_3 x_4 x_5 w_3^4 \end{aligned} \tag{8}$$

Only the terms with even powers of w in the above generating functions remain because only the variables taking the value corresponding to the even powers of w may satisfy the equations. Now let $w_1 = w_2 = w_3 = w$. The total generating function for the above equations is

$$\begin{aligned} g &= g_1 \cdot g_2 \cdot g_3 \\ &= x_1^0 x_2^0 x_3^0 x_4^0 x_5^0 w^0 + x_1 x_2 x_3 x_4 x_5 w^5 \end{aligned} \tag{9}$$

Having interest in the number of solutions of various weight, replace x_i^j with 1, and obtain $g = 1 + w^5$. This result means that the number of solutions with summation of all of the variables equal to 0, is 1 and with the summation of all of the variables equal to 5, is 1. This result can be easily verified by noticing the symmetry of the equations.

In this example, it seems it is not necessary to distinguish various w_i if one first calculates g_1, then g_2, g_3 and lastly, g. However, distinguishing w_i^j gives more freedom to the calculation, allowing us to multiply some of the factors of g_1 with some of the factors of g_2 without any confusion. In some very structured equations, this may save a lot of memory space.

Let us consider a P3 Projection Code with slopes 1, 1/2 and 1/3, one data row (row a), three parity check rows (rows b, c and d) and a block length of 7.

The code block is shown in Fig.3.

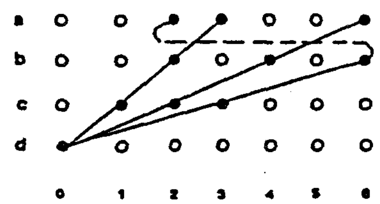


Fig. 3 A P3 Cylindrical Projection Code

To describe it simply, consider the parity check line with slope 1/3, and passing through the point (b, 6) as B6, call the parity check line with slope 1/2 and passing through the point (C, 2) as C2, call the parity check line with slope 1 and passing through the point (d, 0) as D0, and so on. Further, use b_6 to represent the variable at the position (b, 6), c_2 to stand for the variable at the position (c, 2), d_0 to stand for the variable at the position (d, 0) and so on. B6, C2 and B2 stand for linear equations under mod 2. For line D0, the parity check equation is

$$a_3+b_2+c_1+d_0=0 \mod 2 \tag{10}$$

and its generating function is

$$(a_3^0+a_3\cdot w_{D0})(b_2^0+b_2\cdot w_{D0})(c_1^0+c_1\cdot w_{D0})(d_0^0+d_0\cdot w_{D0}) \tag{11}$$

The parity check equation for line C2 is

$$a_6+b_4+c_2+d_0=0 \mod 2 \tag{12}$$

and the generating function is

$$(a_6^0+a_6\cdot w_{C2})(b_4^0+b_4\cdot w_{C2})(c_2^0+c_2\cdot w_{C2})(d_0^0+d_0\cdot w_{C2}) \tag{13}$$

The parity check equation for line B6 is

$$a_2+b_6+c_3+d_0=0 \mod 2 \tag{14}$$

and the corresponding generating function is

$$(a_2^0+a_2\cdot w_{B6})(b_6^0+b_6\cdot w_{B6})(c_3^0+c_3 w\cdot_{B6})(d_0^0+d_0\cdot w_{B6}) \tag{15}$$

Now let us calculate the generating factor of the point d_0. The factor is the product of all the factors which contain the symbol d_0

$$(d_0^0+d_0\cdot w_{D0})(d_0^0+d_0\cdot w_{C2})(d_0^0+d_0\cdot w_{B6})$$
$$=(d_0^0+d_0\cdot w_{D0}\cdot w_{C2}\cdot w^{-1})(d_0^0+d_0\cdot w_{B6})$$
$$=(d_0^0+d0\cdot w_{D0}\cdot w_{C2}\cdot w_{B6}\cdot w^{-2}) \tag{16}$$

Because we will never meet d_0 again, we replace d_0 and d_0^0 with 1 and set $w_{D0}=w_{C2}=w_{B6}=w$. Hence we obtain

$$1+B6C2D0w \tag{17}$$

Generally, for any point (i, j) in the block, we have the corresponding generating factor

$$(1+B_{y(0)}\cdot C_{y(1)}\cdot D_{y(2)}\cdots w) \tag{18}$$

where

$$y(l)=[h+i-(j-l)\cdot spl(l)] \mod h \tag{19}$$

with $spl(l)$ stand for the inverse of the No. l slope.

$$0\le i\le k+r-1;$$
$$0\le j\le h-1; \tag{20}$$
$$0\le l\le r-1;$$

The above expression can be explained intuitively. For example, point d_0 has two possible values, either 1 or 0. If it is 1, then on line B6, there must be another bit which is 1 in order to satisfy the parity check equation. Similarly, there must be at least another bit on line C6 taking 1 and at least one bit on line D0 taking 1. So there must be such a factor as (1+B0.C6.D0.w) in the generating function.

Multiply the generating factors along one parity check line to obtain the product of the factors along this line, one can delete any term which contains the symbol of the line, because after that the same symbol will not be present in any factor. Further, for any slope line symbol, if the symbol is not present in the factors to be calculated, then the terms which contain the line symbol can be deleted. For example, after obtaining the product of the factors corresponding to a_3, b_2, c_1, and d_0, one can delete any term which contains the line symbol D0. Similarly, delete any term which has the line symbol D1 after obtaining the product of the factors of a_3, b_2, c_1, and d_0; delete any term which has the symbol D2 in the expansion of the product of the factors of a_3, b_2, c_1, and d_0; and delete any term which has the line symbol D3 after obtaining the corresponding product. Finally, multiple these four products together and delete any term which has the line symbol B4.

Example

A P3 Projection code with block length 10, one data row, three parity check rows and slopes 1, 1/2, 1/4. Table 1 gives the weight distribution of the codewords. Bounds in the bit error rate are presented in Table 2.

Table 1 The Wight Distribution

Weight	Number
0	1
8	10
10	4
12	150
14	180
16	1205

Table 2 The Lower Bounds and Upper Bounds

Input Error	Lower Bound	Upper Bound
0.1	0.257×10^{-3}	0.421×10^{-1}
0.05	0.807×10^{-4}	0.834×10^{-3}
0.01	0.497×10^{-6}	0.717×10^{-6}
0.001	0.676×10^{-10}	0.700×10^{-10}
0.0001	0.698×10^{-14}	0.700×10^{-14}

IV. DISCUSSION

Due to the fact that not all the solutions of the equations based on the structure of a cylindrical Projection code correspond to codewords of the cylindrical Projection code, it can be shown that for some solutions, the parity check lines are satisfied but the corresponding codes are not codewords. For example, in the situation of P3, suppose two data rows are all 1 and all other rows are all 0. The parity check lines are satisfied but this pattern is not a codeword for P3. However, if the blocklength of a Projection code is long enough, its real weight distribution can be identified with the solution of the equations when only small weights are taken into consideration.

Another question is that when the number of bits in a block code become very large, both the area outside the cycles of Fig.2 and the area in the overlapped rectangles of Fig.1 are very large. In order to obtain tight bounds, the input bit error rate should be small.

V. CONCLUSIONS

The generating functions have been used to find the weight distribution of the Projection Codes and from the weight distribution of that code both the upper bound and lower bound can be obtained.

The generating function has a very strong structure. Therefore, we hope, based on the structure, some much simpler algorithm can be found.

VI. REFERENCES

[1] David Manela, "A new class of forward error correcting codes for burst and random errors," Doctor Dissertation of CUNY, 1987.

[2] Gary R. Lomp, "A new burst & random error correcting code: nonbinary Projection code," Final Report, NSF Award Number ISI-8860421.

[3] H. Taub and D. L. Schilling, "Principles of communication systems," second edition, McGraw-Hill Pub. 1986.

[4] R. E. Blahut, "Theory and practice of error control codes," Addison Wesley Pub. 1983.

[5] Alan Tucker, "Applied combinatorics," Addison Wesley Pub. 1979.

[6] Gregory M. Constantine, "Combinatorial theory and statistical design," John Wiley & Sons, Inc.

CONCATENATED MULTILEVEL CODING

K. Fazel[1] and A. Chouly

Laboratoires d'Electronique Philips
22 avenue Descartes, 94453 Limeil-Brévannes Cedex, France

ABSTRACT

In a recent paper [10], a multistage decoding using erasing technique was studied. It was shown that for PSK and QAM modulation schemes, more than 1 dB of coding gain can be achieved with respect to a non-erased scheme. However, it is very difficult to get higher performances with single codes because of the drastic increase in decoder complexity. An alternative solution consists in employing a concatenated scheme based on a parity check inner code and a Reed Solomon outer code. The outer code performance is strongly influenced by the decoding strategy of the inner code. This may follow a hard detection or a ML soft decoding combined with an erasing procedure. In this paper, different strategies of decoding the inner code applied in the case of a multilevel coding are studied. The first strategy recapitulates the classical method of decoding the concatenated code. The second strategy is based on an erasing technique using a hard detection. The ML detection is applied in strategy 3. The fourth strategy is the combination of the second and the third strategies : using a ML decoding and the erasing technique. For each decoding strategy, the overall performances of the system are derived analytically and the criterion of erasing is defined. The analytical results show that significant coding gain can be achieved by using the erasing technique with a ML Viterbi decoder.

1. INTRODUCTION

On an additive white Gaussian noise (AWGN) channel, studies [1] have shown that trellis-coded modulation (TCM) schemes can provide an asymptotic coding gain of 3 to 5 dB with simple codes. However, in high rate applications, the hardware complexity of the Viterbi decoder required for TCM decoding becomes prohibitive. Therefore, new families of block-coded modulation (BCM) schemes [2] have been proposed. An interesting technique for implementation of BCM schemes is the multilevel coding introduced by Imai and Hirakawa [3-5]. It allows the use of the suboptimal multistage decoding procedures that have performance/ complexity advantages over maximum likelihood (ML) decoding. Afterwards, a large number of theoretical concepts were investigated by many people in this context [6-9].

In a recent paper [10], a multistage decoding using erasing technique was studied. It is shown that for PSK and QAM modulation schemes, more than 1 dB of coding gain can be achieved with respect to a non-erased scheme. The decoding complexity remains nearly the same as a non-erased decoding. However, it is very difficult to get higher performances with single codes, because of the drastic increase in decoder complexity. An alternative solution consists in employing a concatenated scheme in which the coding process consists in using two or more simple codes. Thus, it is a practical means [11] of achieving long blocks or constraint lengths, i.e. achieving a large coding gain with reasonable complexity (operations/bit).

One interesting combination is based on a parity check (PC) inner code and a block Reed-Solomon (RS) outer code. The outer code performance is strongly influenced by the decoding strategy of the inner code. This may follow a hard detection or a ML soft decoding, combined with an erasing procedure.

The purpose of this paper is to study different strategies of decoding the concatenated inner code, applied in the case of a multilevel coding with a multistage decoding procedure.

The paper is organized as follows : Section 2 summarizes the general principle of a concatenated multilevel coding based on Ungerboeck's partitioning. The multistage decoding process employing different decoding strategies is detailed in Section 3. The performance analysis for each decoding strategy, using an 8-PSK modulation is given in Section 4. Section 5 gives simulation results for different decoding procedures by assuming a spectral efficiency of 2.7 bits/s/Hz. It will be shown that considerable coding gain at a bit error rate (BER) of 10^{-11} can be achieved with very simple codes. Finally, Section 6 is for our conclusions.

2. PRINCIPLE OF CONCATENATED MULTILEVEL CODING

An important consequence of partitioning, exploited by Ungerboeck and others, is that coding gain may still result even if some bits mapped to high levels of partitioning are uncoded since they are protected by a large subset distance [1]. Assuming a 2^m-point signal constellation A_0, with minimum Euclidean distance d_0 and able to transmit m bits per symbol, this set can be partitioned into $2^{\tilde{m}}$ ($\tilde{m} \leq m$) distinct subsets S_j, $j = 1, \ldots, 2^{\tilde{m}}$. At each partition level, the minimum intra-subset Euclidean distance d_j satisfies the following inequalities :

$$d_0 \leq d_1 \leq \ldots \leq d_{\tilde{m}}$$

Hence, the $\tilde{m}$ bits b^j, $j = 1, \ldots, \tilde{m}$ (where b^j represents the bit mapped to the j^{th} level of partitioning) are mapped to the $2^{\tilde{m}}$ subsets and the $(m - \tilde{m})$ remaining bits select a signal point in this subset. This process of mapping gives the classification of m bits with different vulnerabilities to the channel noise. This fact implies the use of different levels of coding with appropriate protection capacities instead of one level of coding for all partition bits. This represents the principle of multilevel coding which consists of utilizing $\tilde{m}$ different codes

[1] German Aerospace Research Est., 8031 Oberpfaffenhofen, Germany

 CH3132-8/0000-0329 $3.00 © 1992 IEEE

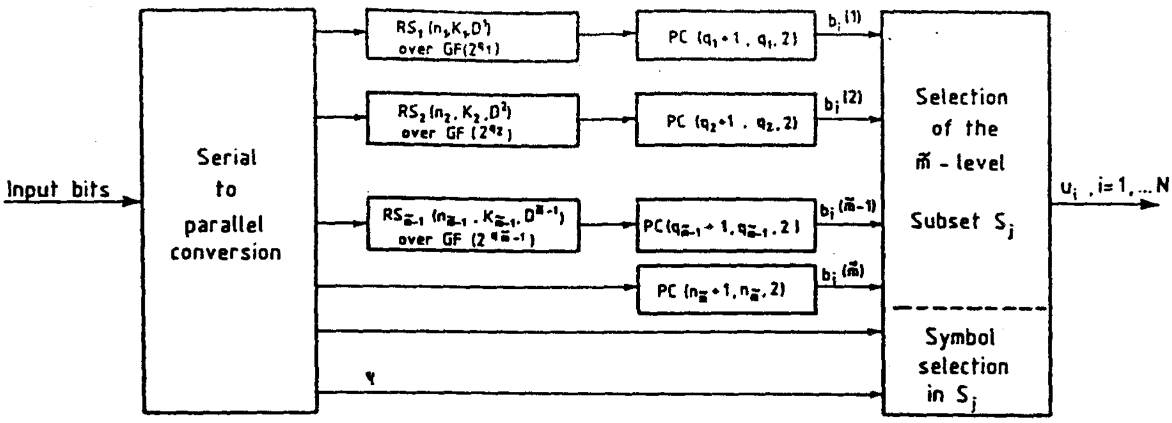


Fig. 1 : Principle of concatenated multilevel coding.

with different levels of protection, to encode the corresponding bit b^j, $j = 1, ..., \tilde{m}$.

The squared minimum Euclidean distance of the multilevel coding verifies [5] :

$$d^2 = \min(\delta_j d_{j-1}^2) ; j = 1, ..., \tilde{m}+1, \text{ with } \delta_{\tilde{m}+1} = 1$$

where δ_j is the Hamming distance of the code for the j^{th} partition level.

Consequently, the asymptotic coding gain of a ML detection is given by :

$$G_a(dB) = 10 \log \left(d^2 R / d_0^2\right)$$

where R is the total coding rate.

Hence, to obtain a high coding gain, the multilevel coding is optimized if :

$$\delta_1 \geq \delta_2 ... \geq \delta_{\tilde{m}}$$

To assume large δ_j (especially large δ_1), one can use a single code with a powerful error correcting capacity, which results in an increase of decoder complexity. An alternative solution is to use two or more simple codes instead of one guaranteeing the same distance δ_j. In the latter, it should be noted that the issued coding distance δ_j is the product of several (l) concatenated codes distances D_l^j :

$$\delta_j = D_1^j \times D_2^j \times ... \times D_l^j$$

So, to achieve large values of δ_j, it is important to choose an appropriate value for D_l^j, i.e. the appropriate simple codes.

One interesting combination is based on $l = 2$ concatenated codes : a PC inner code and a RS outer code.

Fig. 1 illustrates the principle of the scheme : K information bits are demultiplexed into m blocks K_j, $j = 1, ..., m$ and $\tilde{m}$ blocks are encoded. The $(\tilde{m}-1)$ first blocks are coded by powerful concatenated codes (which correspond to the more vulnerable bits). The outer codes are the well known $RS_j(n_j, k_j, D^j)$ codes over a Galois Field GF (2^{q_j}), where $K_j = k_j q_j$ $(j = 1, ..., \tilde{m}-1)$. Then each symbol ($q_j$ bits) of RS_j is coded by a PC code $E_j(q_j+1, q_j, 2)$, $j = 1, ..., \tilde{m}-1$. So the corresponding Hamming distance δ_j is $2D^j$. The last block $j = \tilde{m}$ is coded only by the PC code $E_{\tilde{m}}(n_{\tilde{m}}+1, n_{\tilde{m}}, 2)$, where $K_{\tilde{m}}$ is a multiple of $n_{\tilde{m}}$, i.e. $K_{\tilde{m}} = h \times n_{\tilde{m}}$.

Let $N_j = (q_j+1)\, n_j = h(n_{\tilde{m}}+1) = N$, $j = 1, ..., \tilde{m}-1$ (which we assume in the sequel), the process of coding can be described by a matrix structure as used in the design of BCM.

The coded bits $(j = 1, ..., \tilde{m})$ and uncoded bits $(j = \tilde{m}+1, ..., m)$ at level j are denoted by b_i^j, $i = 1, ..., N$.

The choice of RS and PC codes is based on the small decoding complexity :

- The symbol of the RS code is the information part of the inner code. Errors in the inner codeword affect only one symbol of the RS code. So no interleaving is needed between these two codes ;
- To decode each level, only one RS decoder circuit is needed (if E_j is over the same GF), since the decoding process is performed step by step.

The outer code performance is strongly influenced by the decoding strategy of the inner code.

3. CONCATENATED MULTISTAGE DECODING STRATEGIES

It is well known that the optimum decoding strategy is maximum-likelihood decoding (MLD). This involves correlating the received waveform with each of the $2^{\Sigma K_j}$ waveforms corresponding to the codeword of signal space. Generally, as the value of $\sum K_j$ is very large, this method becomes prohibitively complex. A suboptimum decoding method called multistage decoding [8] having a better performance/ complexity trade-off than MLD can be performed.

Let $U = (U_1, ..., U_N)$ be the transmitted block of N symbols (corresponding to b_i^j, $j = 1, ..., m, i = 1, ..., N$). Since the channel is corrupted by noise, the received block will be $\rho = (\rho_1, ..., \rho_N)$. From ρ, we try to retrieve the transmitted bits b_i^j , $j = 1, ..., m$, $i = 1, ..., N$.

The decoding process is performed by a successive estimation of $b_i^1, b_i^2, ..., b_i^{j-1}$. The estimate of b_i^j indicated by $\hat{b}_i^j$, $i = 1, ..., N$ is carried out by using ρ and $\hat{b}_i^1, \hat{b}_i^2, ..., \hat{b}_i^{j-1}$.

Since we assume the same PC inner code and the same codeword length for the outer code (of course with different code rates), for $(\tilde{m}-1)$ levels, we need only one RS and one PC decoder (Fig. 2). The RS decoder has different correction capacities, which can correct up to $t_1 = \lfloor (D^1 - 1)/2 \rfloor$ symbol errors.

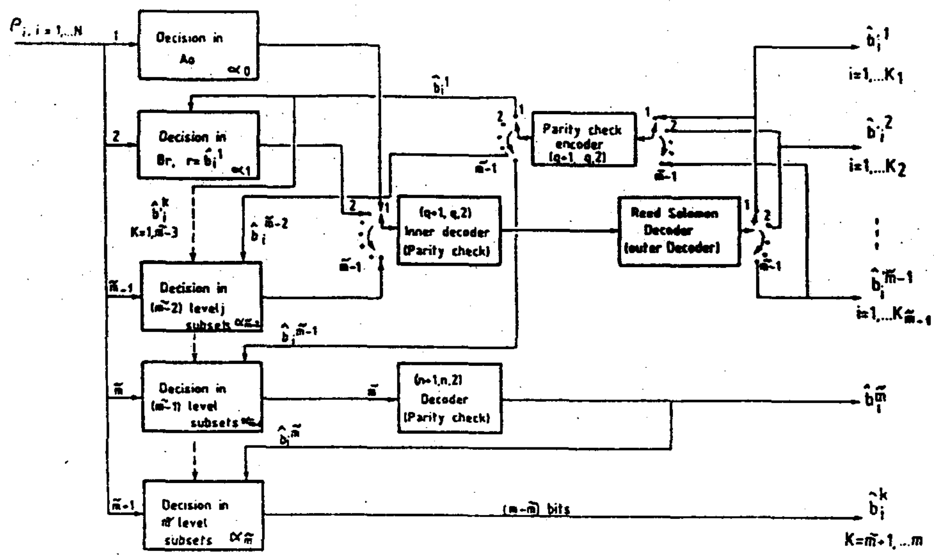


Fig. 2 : The concatenated multistage decoder.

For the $\bar{m}^{th}$ level, we need only a parity check decoder. For each level $j, j = 1, \dots, \bar{m}+1$, a different hard detector α_{j-1} is needed.

To estimate b_i^j from ρ, a first estimation is carried out by α_{j-1}, by using $\hat{b}_i^1, \dots, \hat{b}_i^{j-1}$. Then the concatenated decoding (inner and outer decoders) gives the final estimate.

Different strategies of decoding can be performed, in which we use the same RS outer decoder which can correct errors and fills the erasures [11] :

- α_j is a hard decision estimator and the inner decoder is an error detector,
- α_j is a hard/erasing decision estimator and the inner decoder is an error detector and it can fill one erasure [10],
- α_j is a hard decision estimator combined with a ML Viterbi inner decoder,
- α_j is combined with a soft/erasing Viterbi inner decoder.

Without loss of generality to describe in more detail the decoding process for each strategy, let's take an example of 8-PSK modulation with two coded partitioning levels ($\bar{m} = 2$). Three estimators $\alpha_0, \alpha_1, \alpha_2$, two inner decoders and one error-erasure RS decoder are needed.

A/ First strategy

This first strategy is the classical method of decoding a concatenated multilevel code :

- To estimate b_i^1, $i = 1, \dots, N$, mapped to the first level of partition, a hard decision is firstly carried out by α_0 in A_0 (8-PSK) for all received symbols ρ_i, $i = 1, \dots, N$. A first estimate of b_i^1, $i = 1, \dots, N$, noted $\tilde{b}_i$, $i = 1, \dots, N$ is obtained. Then the inner error-detector is performed for every $(q_1 + 1)$ estimated bits $\tilde{b}_i$. If an error is detected, q_1 bits (the symbol of RS outer code) are erased. Finally, the RS decoder is used and gives the final estimation. At the output of the RS decoder, we dispose of only $n_1 q_1$ bits ($\hat{b}'^1_i$, $i = 1, \dots, n_1 q_1$). The missing bits (n_1 bits) are the parity check bits of the inner code, which were dropped before the outer decoding. Then the $\hat{b}_i^1$, $i = 1, \dots, N$ bits are estimated by re-encoding the symbols of the RS decoder.

- To estimate b_i^2, $i = 1, \dots, N$, mapped to the second partition level, a second hard-detection is carried out by α_1 in the first-step partitioning subset $B_r (r = \hat{b}_i^1)$ for all received symbols ρ_i, $i = 1, \dots, N$. A first estimation of b_i^2, $i = 1, \dots, N$ is obtained. Following this strategy, since this is only coded by a $PC(n + 1, n, 2)$, just one error is detectable in every $(n + 1)$ bits, but non correctable since there is no outer code. We will immediately obtain the final estimation $\hat{b}_i^2$, $i = 1, \dots, N$.

- Finally, to estimate the last remaining bits b_i^3, $i = 1, \dots, N$, we perform a hard-detection α_2 in the second step partitioning subsets $C_{q_i} (q_i = \hat{b}_i^1 + 2\hat{b}_i^2)$ over all received symbols ρ_i, $i = 1, \dots, N$. Since there is no coding, we immediately have the final estimate $\hat{b}_i^3$, $i = 1, \dots, N$.

B/ Second strategy

This strategy of decoding is based on a hard/erasing decision of α_j [10]. To increase the reliability of the first hard-decision estimators α_{j-1} $(j = 1,2)$, we erase the estimated bits $\tilde{b}_i^j$ of α_{j-1}, if ρ_i falls within the interval $l \pm \psi_j$ (l is the hard-decision threshold in the $(j-1)^{th}$-step partitioning subsets of A_0). In this case, α_{j-1} provides three states "0", "1", and "X" instead of two, where X represents the erased bits. Then the inner decoder will fill the erasure. In the case of an 8-PSK, the decoding process is the following :

- ***Estimation of bits b_i^1, $i = 1, \dots, N$, mapped to the first level of partition*** :

A hard-decision with erasing is firstly carried out by α_0 in A_0 for all received symbols ρ_i, $i = 1, \dots, N$. A first estimate of b_i^1, $i = 1, \dots, N$, noted $\tilde{b}_i^1$, $i = 1, \dots, N$ is obtained. $\tilde{b}_i^1$, $i = 1, \dots, N$ is erased if θ_i^1, the estimated angle between the received symbol ρ_i and its nearest point in A_0 satisfies these inequalities :

$$\frac{\pi}{8} - \Psi_1 < \theta_i^1 \leq \frac{\pi}{8} \qquad (\Psi_1 = 0 \Rightarrow \text{non-erasing})$$

Thus, α_0 gives three states "0", "1" and erased state "X" : $\tilde{b}_i^1 \in \{0,1,X\}$.
Then the inner decoding is performed for every (q_1+1) estimated $\tilde{b}_i^1$. If one erasure is detected, the inner decoder fills it. If one error in (q_1+1) bits or more than one erasure is detected, q_1 bits (the symbol of RS outer code) are erased. Then the RS outer code is decoded as explained in the first strategy.

– Estimation of bits b_i^2, $i = 1, \ldots, N$, mapped to the second level of partition :

By using $\hat{b}_i^1$, $i = 1, \ldots, N$, a second hard-decision with erasing is carried out by α_1 in $B_r (r = \hat{b}_i^1)$ for all received symbols ρ_i, $i = 1, \ldots, N$. A first estimate of $\tilde{b}_i^2$, $i = 1, \ldots, N$ is obtained. Then $\tilde{b}_i^2$ is erased if θ_i^2, the estimated angle between the received symbol ρ_i and its nearest point in B_r satisfies these inequalities :

$$\frac{\pi}{4} - \Psi_2 < \theta_i^2 \leq \frac{\pi}{4} \qquad (\Psi_2 = 0 \Rightarrow \text{non-erasing})$$

Thus, $\tilde{b}_i^2 \in \{0,1,X\}$; $i = 1, \ldots, N$. Then the erasure PC decoder (which fills one erasure) is carried out over every $(n+1)$ bits $\tilde{b}_i^2$ which gives the final estimation $\hat{b}_i^2$, $i = 1, \ldots, N$.

The final estimate of the remaining uncoded bits b_i^3, $i = 1, \ldots, N$, is the same as explained in the first strategy.

It should be noted that the optimum values of Ψ_j, $j = 1,2$ for the above example are $\Psi_1 \simeq 4°$ and $\Psi_2 \simeq 10°$ [10].

C/ *Third strategy*

In this strategy, we exploited the fact that the codewords of a PC code can be represented by a 2-state trellis. So we can easily perform a soft Viterbi decoding to estimate the PC codewords (Fig. 3). So, for every coded level, the hard decision estimator α_{j-1} can be combined with the Viterbi decoder. The α_{j-1} estimates the two nearest points of the received symbols ρ_i, $i = 1, \ldots, N$, in the subsets and gives their Euclidean distances to the Viterbi decoder. The Viterbi decoder gives q_1 bits : the symbols of the RS error-correcting outer decoder. Then the other processes of decoding are similar to the first and second strategies.

D/ *Fourth strategy*

In the second strategy, we exploit the presence of the ambiguity of decision of α_{j-1} and introduced the notion of erasing to increase its reliability. On the other hand, the third strategy is based on a ML Viterbi decoding.

From these observations, a question arises : "is it possible to introduce the notion of erasing in the Viterbi decoder, in order to increase the reliability of its decision ?" The answer to this question is yes, since an ambiguity exists in each ACS (Addition- Comparison-Selection) of the Viterbi decoder.

So, this last strategy is the combination of the second and third strategies, i.e. at the output of the Viterbi decoder, the symbols are erased (for the concatenated level) if the decision within the Viterbi decoder is ambiguous.

Fig. 3 illustrates the trellis of a PC $(q_j+1, q_j, 2)$ code. Each branch of this trellis is affected by the Euclidean distance, issued by α_{j-1}. In each state, the Viterbi decoder selects the path having the minimum distance. For instance, as Fig.3 shows, if the difference of the cumulated squared distances $(d_j^2)_A$, $(d_j^2)_{A'}$ of two paths (A) and (A') respectively, is small : $|(d_j^2)_A - (d_j^2)_{A'}| \leq \varepsilon_r$, the decision within the Viterbi decoder is ambiguous.

In this strategy, the erasing technique is introduced after the Viterbi decoding, if the chosen path contains an ambiguity.

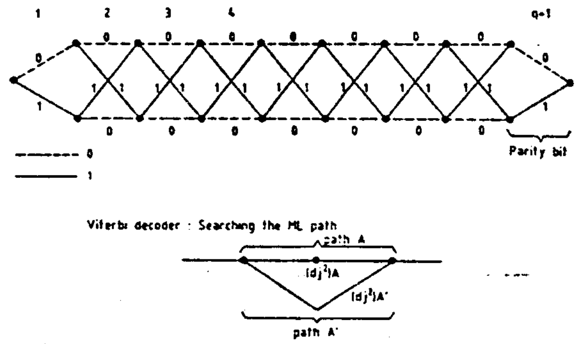


Fig. 3 : Trellis of the parity check code $(q+1,q,2)$.

In the next section, we will examine how to find the optimal value of ε_r and to evaluate the performances of the different decoding strategies.

4. PERFORMANCE EVALUATION

The overall performance of the system depends on the respective performance of each level of coding. This performance can be measured by the BER at the output of the system for a given signal to noise ratio (SNR). For each level j, we derive the bit error rate P_{ej}. The BER is the mean value of P_{ej} which is upper bounded by :

$$BER \leq \frac{1}{(m-\tilde{m})N + \sum_{j=1}^{\tilde{m}} K_j} \left(\sum_{j=1}^{\tilde{m}} K_j P_{ej} + N \sum_{j=\tilde{m}+1}^{m} P_{ej} \right)$$

For the concatenated coded levels, P_{ej} is the BER at the output of the RS decoder. For a RS (n_j, k_j, D^j) code which can correct up to v symbol errors and can fill ω erasures, and in the worst case adds no more than t_ω errors, P_{ej} is given by :

$$P_{ej} \leq \frac{1}{q_j n_j} \sum_{2v+\omega \geq D^j} (t_\omega + v) P_j(v,\omega), \; j = 1, \ldots, \tilde{m}-1 \quad (1)$$

where :

$$t_\omega = \left\lfloor \frac{D^j - \omega - 1}{2} \right\rfloor$$

and :

$$P_j(v,\omega) = \binom{\omega}{n_j}\binom{v}{n_j-\omega} P_j^v Q_j^\omega (1-P_j-Q_j)^{n_j-v-\omega}$$

where P_j and Q_j are respectively the error and erased symbol rates at the output of the inner decoder.

For the single coded level ($j=\bar{m}$), P_{ej} is the probability of error at the output of the inner decoder.

For $j=\bar{m}+1, \ldots, m$, P_{ej} depends mainly on the probability of detection in the $\bar{m}$-step partitioning subsets of A_0.

So, the system performance depends strongly on P_{ej}, i.e. P_j and Q_j. In each strategy, we try to minimize P_{ej}. Let us take the above example to compute P_{ej} for different strategies in an AWGN channel.

A/ First strategy

In this strategy, for the first level of partition, the expression for the symbol-error rate P_1 and the symbol-erased rate Q_1 are given by :

$$P_1 \simeq \binom{q+1}{2} erfc^2(\sigma \sin \pi/8)$$

$$Q_1 \simeq (q+1)\, erfc(\sigma \sin \pi/8)$$

with $\sigma^2 = R/N_0$ (N_0 is the one sided noise spectral density). P_{e1} is given by (1) and the expressions of P_{e2} and P_{e3} (Bayes'rule) are given by :

$$P_{e2} \simeq \frac{1}{2} P_{e1} + erfc(\sigma\sqrt{2}/2)(1-P_{e1})$$

$$P_{e3} \simeq \frac{1}{2} P_{e2}(1-P_{e1}) + \frac{1}{2} erfc(\sigma)(1-P_{e2})(1-P_{e1})$$

B/ Second strategy

The computation of P_{ej} for this strategy is given in [10]. We recapitulate briefly the results :

$$P_1 \simeq \binom{q+1}{2} erfc^2(\sigma \sin \Phi_1)$$

$$Q_1 \simeq \binom{q+1}{2}\left(erfc(\sigma \sin \Phi'_1) - erfc(\sigma \sin \Phi_1)\right)^2 + (q+1)\, erfc(\sigma \sin \Phi_1)$$

with $\Phi_1 = \frac{\pi}{8} + \Psi_{1opt}$, $\Phi'_1 = \frac{\pi}{8} - \Psi_{1opt}$, and $\Psi_{1opt} \simeq 4°$.

P_{e1} is derived by using the expression (1), and the expressions of P_{e2} and P_{e3} (Bayes'rule) are given by :

$$P_{e2} \simeq \frac{1}{2} P_{e1} + \left[\binom{n+1}{2}\left(erfc(\sigma \sin \Phi'_2) - erfc(\sigma \sin \Phi_2)\right)^2 + (n+1)\, erfc(\sigma \sin \Phi_2)\right](1-P_{e1})$$

with $\Phi_2 = \frac{\pi}{4} + \Psi_{2opt}$, $\Phi'_2 = \frac{\pi}{4} - \Psi_{2opt}$, and $\Psi_{2opt} \simeq 10°$.

$$P_{e3} \simeq \frac{1}{2} P_{e2}(1-P_{e1}) + \frac{1}{2} erfc(\sigma)(1-P_{e2})(1-P_{e1})$$

C/ Third strategy

The inner decoder follows the principle of MLD decoding, hence the expression of P_1 is given by (non-erasing) :

$$P_1 \simeq q(q+1)\, erfc\left(\sigma\sqrt{2} \sin \pi/8\right).$$

P_{e1} is given by (1) and P_{e2} and P_{e3} are given also by :

$$P_{e2} \simeq \frac{1}{2} P_{e1} + 2n\, erfc(\sigma)(1-P_{e1})$$

$$P_{e3} \simeq \frac{1}{2} P_{e2}(1-P_{e1}) + \frac{1}{2} erfc(\sigma)(1-P_{e2})(1-P_{e1})$$

Hence the squared minimum equivalent distance of the first level is $2d_0^2(t+1)$ where t is the error correcting capacity of the outer code.

Since the second level is coded only by a PC code and there is no coding in the third level, the squared minimum distance of the system is given by :

$$(d^2_{min})_3 = \min\left[2d_0^2(t+1),\ 2d_1^2,\ d_2^2\right] = \min\left[2d_0^2(t+1),\ 4\right]$$

with $d_0 = 2\sin(\pi/8)$.

D/ Fourth strategy

As we have seen, the Viterbi decoder chooses the path in the trellis having the minimum distance. But, when the difference between two cumulated distances is small (less than ε_p for the p^{th} level), the decision of choosing the correct path will not be very reliable and will be erased.

Thus, for the first level of partition, the expression of symbol error rate P_1 and symbol erased rate Q_1 are given by:

$$P_1 \simeq q(q+1)\, erfc(d'\sigma/2)$$

$$Q_1 \simeq q(q+1)\left[erfc(d''\sigma/2) - erfc(d'/2\sigma)\right]$$

with $d' = \sqrt{2}\, d_0 + \dfrac{\varepsilon_1}{\sqrt{2}\, d_0}$, and $d'' = \sqrt{2}\, d_0 - \dfrac{\varepsilon_1}{\sqrt{2}\, d_0}$.

For high SNR, the BER at the output of the outer decoder is given by the two terms $P_1^{(t+1)}$ or $Q_1^{2(t+1)}$.

Hence, the squared minimum equivalent distance of the first level is :

$$(t+1)\min\left[d'^2, 2d''^2\right]$$

This distance is maximal if $d'^2 = 2d''^2$.

From this expression, we can easily derive the optimal value of ε_1 :

$$\varepsilon_{1opt} = 2d_0^2(3-2\sqrt{2}) = 0.201$$

For this value of ε_1, the coding gain with respect to a non-erased scheme can be easily derived :

$$G = 10\log\left(\frac{d^2(\varepsilon_{1\,opt})}{d^2(\varepsilon_1=0)}\right) = 1.37\ \text{dB}$$

The expressions of P_{e2} and P_{e3} are the same as the third strategy.

Finally, the squared minimum Euclidean distance for this strategy can be derived by :

$$(d^2_{min})_4 = \min [(t+1) d^2(\varepsilon_{1opt}), 4]$$

5. EXAMPLES

The performances of the following concatenated code using an 8-PSK modulation, for different decoding strategies, are derived :

- First level: the inner code is a PC (9,8,2) code and the outer code is a RS(40,34,7) code defined over GF (2^8),
- Second level: this level is coded only by a PC (20,19,2) code,
- And the third level is uncoded.

In Fig. 4, the bit error rate curves versus E_b/N_0 for the above strategies are plotted. These curves show that for BER = 10^{-11}, the second strategy provides a coding gain of 0.9 dB with respect to the first strategy. Although the third strategy is based on a ML decoding, it has nearly the same performance as the second strategy for BER = 10^{-11}, since the third strategy has a higher error coefficient. But the fourth strategy provides 0.6 dB of coding gain with respect to the third strategy.

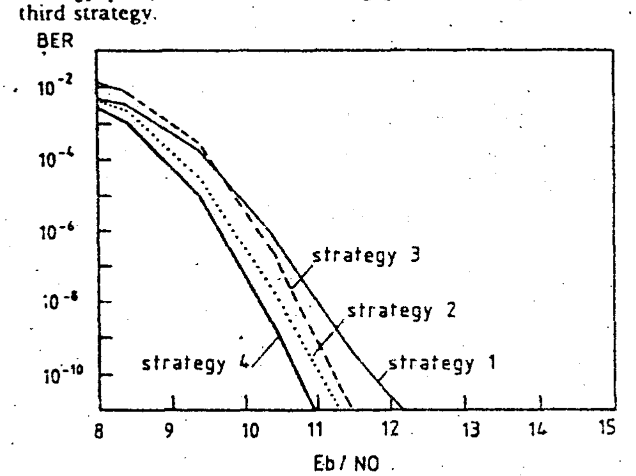


Fig. 4 : The performances of different decoding strategies.

These results show that the erasing technique applied with different strategies of decoding of a concatenated multilevel coding performs well. The decoding complexity is similar to a non-erased decoding.

6. CONCLUSIONS

In this paper, we have developed different decoding techniques for concatenated multilevel coding schemes. For each decoding strategy, the performances of the system have been derived analytically. The results have shown that on an AWGN channel, significant coding gain can be achieved with a ML Viterbi decoder using the erasing technique. This improvement is obtained without increasing the decoder complexity with respect to the non-erased decoding technique. Note that the criterion of erasing can be adapted for particular channels such as the Rayleigh fading channel.

REFERENCES

[1] G. Ungerboeck, "Channel coding with multilevel/phase signals", *IEEE Trans. Inform. Theory*, vol. IT-28, n° 1, pp. 55-67, January 1982.

[2] A. Chouly and H. Sari, "Design and performance of block coded modulation for digital microwave radio systems" *IEEE Trans. Commun*, vol. Com-38. pp. 727-733, May 1990.

[3] H. Imai and S. Hirakawa, "A new multilevel coding method using error-correction codes", *IEEE Trans. Inform. Theory*, vol. IT-23, n° 3, pp. 371-377, May 1977.

[4] E. L. Cusack, "Error control codes for QAM signalling", *Electron. Lett.*, vol. 20, pp. 62-63, January 1984.

[5] S. I. Sayegh, "A class of optimum block codes in signal space", *IEEE Trans. Commun.*, vol. COM-34, n° 10, pp. 1043-1045, October 1986.

[6] G. J. Pottie and D. P. Taylor, "Multilevel codes based on partitioning", *IEEE Trans. Inform. Theory*, vol. IT-35, n° 3, pp. 87-98, January 1989.

[7] A. R. Calderbank and N. J. A. Sloane, "New trellis codes based on lattices and cosets", *IEEE Trans. Inform. Theory*, vol. IT-33, n° 2, pp. 177-195, March 1987.

[8] A. R. Calderbank, "Multilevel codes and multistage decoding", *IEEE Trans. Commun.*, vol. 37, n° 3, pp. 222-229, March 1989.

[9] R. M. Tanner, "Algebraic construction of large euclidean distance combined coding modulation system", *IEEE Int. Symp. on IT*, Ann Arbor, MI, October 6-9, 1986.

[10] K. Fazel and A. Chouly, "Multistage decoding using erasing technique for multilevel coding", *GLOBECOM'91 Conf. Rec.*, vol. 2, pp. 779-784, December 1991, Phoenix, AZ.

[11] R. E. Blahut, "Theory and practice of error control codes", Addison Wesley Publishing Company, 1984.

Algebraic Signal Processing in Truncated p-adic Arithmetic for Linear Channels with Memory

M. Hassner[1], G. Fettweis[2], K. Abdel-Ghaffar[3], C.J. Williamson[4]

[1] IBM Almaden Research Center, San Jose, CA 95120-6099
[2] Teknekron Communications, 2121 Allston Wy, Berkeley, CA 94704
[3] University of California, Davis, EE and CS Dept., Davis, CA, 95616
[4] Cal Poly State U., Mathematics Dept., San Luis Obispo, CA, 95616

Abstract

We consider bandlimited linear channels with finite memory which is formally described by a polynomial $g(z)$ with integer coefficients in the variable z that represents the unit delay operator. We present a signal processing method that operates on blocks of finite precision samples of the channel output. This signal processing method is a "soft-decision" algebraic decoder that operates in real truncated p-adic arithmetic in which computations are exact i.e. there is no no round-off error. It furthermore lends itself to an efficient digital implementation whose complexity is a linear function of the memory size.

1 Introduction

We consider bandlimited linear channels with finite memory which is formally described by a polynomial $g(z)$ with integer coefficients in the variable z that represents the unit delay operator. The roots of the polynomial $g(z)$, which generally are complex algebraic integers, determine a finitely generated algebraic number field K.

We imbed this algebraic number field into a p-adic number field Q_p in which the memory polynomial $g(z)$ factors. The zeroes of $g(z)$ in Q_p are p-adic integers. The channel output is divisible by $g(z)$ and hence in the absence of noise will have zeroes at the same locations and of the same order as $g(z)$ in Q_p.

We present a signal processing method that operates on blocks of finite precision samples of the channel output. These numbers are transformed and used to compute syndromes which are the values of the output p-adic function as well as its derivatives of appropriate order at the locations of the zeroes of the divisor $g(z)$. The syndromes are then used in rational Hermite interpolation formulas, computable in terms of generalized Vandermonde determinants, to provide error pattern estimates which are subtracted from the output samples. This signal processing method is a "soft-decision" algebraic decoder that operates in real truncated p-adic arithmetic in which computations are exact i.e. there is no no round-off error. It furthermore lends itself to an efficient digital implementation whose complexity is a linear function of the memory size.

2 Channel Model

We consider bandlimited channels with finite memory m whose output samples $\{y_k\}$ are determined by the inputs $\{x_k\}$ in the following manner

$$y_{k+r} = \sum_{j=1}^{r} g_{r-j} x_{k+r-j} + x_{k+r}$$

The r coefficients $\{g_{r-j}, j = 1, \ldots, r\}$ are assumed to be integers. If the unit delay operator is described by the variable z, then the channel operation on the inputs $\{x_k\}$ can be described in terms of the polynomial $g(z)$

$$g(z) = z^k + \sum_{j=1}^{r} g_{r-j} z^{r-j}$$

Generally the polynomial $g(z)$ does not have integer roots, its roots $\{\omega_i\}_1^s$ are complex algebraic integers, each of which may have multiplicity σ_i as follows

$$g(z) = \prod_{i=1}^{s} (z - \omega_i)^{\sigma_i}$$

We will refer to $\{\omega_i\}_1^s$ as to the null frequencies of the channel and to $r = \sum_1^s \sigma_i$ as to the null order. The distinct null frequencies $\{\omega_i\}_1^s$ extend the field Q of rational numbers into a finitely generated algebraic number field K. The channel output $Y(z) = \sum_k x^k z^k$ has the zero divisor

$g(z)$. This effectively means that the channel acts as an algebraic encoder defined over K. However, the arithmetic over which this encoder is operating prohibits an efficient use of this structure. This motivates us to consider the imbedding of the number field K into a p-adic number field Q_p.

The p-adic number fields were introduced by Hensel [1]. For a fixed prime p, algebraic integers such as the null frequencies $\{\omega_i\}_1^s$, can be described as power series expansions in p where the coefficients are integers in the set $\{0, 1, ..., p-1\}$. For our channel model the admissible primes p for which we obtain an embedding of K are those for which the null frequencies ω_i become units in Q_p [C]. Thus for each i, $1, ..., s$ there exists

$$\omega_i^{p-1} \equiv 1 (mod p)$$

The computation of the zeroes of the channel polynomial $g(z)$ in the field Q_p, in which it has been imbedded, can be performed with any desired accuracy mod p^l through use of Hensel's Lemma [2].

Of particular interest are channel memory polynomials $g(z)$ that have roots which are roots of unity [3]. The p-adic functions divisible by such a polynomial have zeros that recur periodically [4]. The periods, of the form $\{N | N = rm + k, r = 0, ... k-1, m \in Z\}$, will constitute natural codeword boundaries for the stream of transformed channel sample outputs.

3 Truncated p-adic Arithmetic

The output of the bandlimited channel is sampled and quantized. The result is a stream of rational numbers whose numerator and denominator are bounded. The signal processor that operates on these numbers will make roundoff errors. The use of truncated p-adic arithmetic as a remedy to this problem has been suggested by Krishnamurthy [5]. The theoretical basis for doing exact computations in this arithmetic is the strong triangle inequality satisfied by the p-adic metric. A rigorous treatment can be found in Dittenberger [6] where forward and backward maps between the rational and the p-adic number domains are defined. These maps provide the means to transform the rational numbers at the channel output into p-adic numbers, perform computations in Q_p, and then map the result back into the rational domain. A major question is the meaning of these computations. In [6] a Lemma is stated which provides a condition for the one-to-one correspondence between computations with rational and with truncated p-adic numbers.

If the output of the quantizer consists of rational numbers whose numerator has absolute value bounded by an integer M and their denominator has absolute value bounded by the integer N, a range we denote by Q^-, then a necessary and sufficient condition for the existence of a one-to-one correspondence between computations with these numbers and p-adic numbers whose mantissa is of length l and whose exponent is an integer n is as follows

$$2NM < p^{(l+|n|)}$$

This condition has to be satisfied at the output of the p-adic signal processor and may be violated during intermediate computations. It is possible to remap the quantizer arithmetic range into a constrained set of positive integers which are represented as p-adic integers, an assumption we shall make in the sequel.

The use of truncated p-adic arithmetic not only allows for roundoff error-free computation but also lends itself to an efficient implementation. Initial work on the subject can be found in [7], where the arithmetic unit is defined algorithmically and where furthermore a Fast Fourier Transform is introduced. The operations with complex numbers have p-adic counterpart. Thus for example if we use p-adic numbers in base 5 and constrain ourselves to mantissas of length 3, the 4-th roots of unity are represented as follows

$$1 \rightarrow (1, 0, 0), -1 \rightarrow (4, 4, 4), i \rightarrow (2, 1, 2), -i \rightarrow (3, 3, 2)$$

The operations of rational arithmetic are defined for these numbers and signal processing structures such as convolution, harmonic decomposition, etc. can be efficiently implemented.

4 Algebraic Signal Processor

Given distinct null-frequencies $\{\omega_i\}_1^s$ of total null-order r which are imposed by the channel memory $g(z)$, the p-adic Fourier transforms of the sequence of channel output samples at these nulls are

$$S_i = 1/p^r \sum_{k=1}^{N} y_k \omega_i^k$$

The higher order syndromes $S_i^{(t_i)}$ are the Hasse derivatives of S_i written as the sums

$$S_i^{(t_i)} = 1/p^r \sum_{k=1}^{N} \binom{k}{t_i} y_k \omega_i^{k-t_i}$$

In the absence of any noise these syndromes are all zero. Provided there are errors they will manifest themselves through nonzero values of $S_i^{(t_i)}$, where $i = 1, ..., s$ and $t_i = 1, ..., \sigma_i$. The subscript i indexes the distinct zero location whereas the superscript t_i indexes the order of the derivative at the zero location. There exists a unique p-adic polynomial $P(z)$ determined by the syndrome data which is computed by Hermite interpolation as follows [8]

$$P(z) = \sum_i \sum_{t_i} S_i^{(t_i)} \sum_\lambda \frac{\Delta_{it_i,\lambda} z^{\lambda-1}}{\Delta}$$

In this formula Δ is the determinant of a matrix M whose typical entry is $((\lambda-1)!/(\lambda-t_i)!)\omega_i^{\lambda-t_i}$, $\lambda = 1, ..., r$ and the cofactor of a typical entry is $\Delta_{it_i,\lambda}$. The columns of this matrix are indexed by the pairs (i, t_i) lexicographically ordered and the rows by λ. Its determinant Δ is a generalized Vandermonde determinant $\Delta = \prod_{(i,t_i)}((t_i-1)!\prod_{j=1}^{i-1}(\omega_i-\omega_j)^{\sigma_j})$

The existence of this interpolation formula depends on the conditions that the zeros ω_i have to satisfy and which are stated as follows $|\omega_i|_p \leq p^{(-1/(p-1)+\theta)}$ for some θ greater than 0, and $min_{h\neq k}|\omega_k - \omega_h|_p \geq p^{-\delta_k}$. The polynomial $P(z)$ computed by this interpolation formula satisfies $P^{(t_i)}(\omega_i) = S_i^{(t_i)}$ for all (i, t_i). To exploit the existence of this interpolation formula for the purpose of constructing an algebraic decoder we have to formulate a rational Hermite interpolation problem.

For this purpose we adopt the viewpoint of Goppa [9], by which the stream p-adic integers, obtained as samples of the channel output, are viewed as residues of rational functions whose poles belong to a finite set L and which are furthermore divisible by the fixed polynomial $g(z)$ for which we assume that its null frequencies are all roots of unity.

The set of possible poles L is the range of p-adic integers allowed by the quantizer minus the zeros of the polynomial $g(z)$. The cardinality of the set L, N, is the maximum codeword size and the parity check matrix H is defined as the interpolation basis for the linear vector space of all rational (L, g) functions in a manner analogous to Goppa's definition [9]. The syndromes $S_i^{(t_i)}$ are obtained by evaluating these functions and their derivatives, given the received residues, at the zeros of $g(z)$. The rational Hermite interpolation problem is defined as the computation of a rational function $e(z) = p(z)/q(z)$ such that $S_i^{(t_i)} = e^{(t_i)}(\omega_i)$, where the degrees of the polynomials $p(z)$ and $q(z)$ are l and m respectively, $l < m$, $l + m = r$. Recursive algorithms for solving such an interpolation problem in terms of divided differences are reported in the literature [10]. Effectively the solution of the rational interpolation problem is equivalent to the solution of the Key Equation [11] for an algebraic decoder.

The main question is what are the constraints on the values l and m, $l+m = r$, for which there exists a solution, i.e. how many errors can the described interpolation decoder correct. Because we attach linear codes with linear spaces of rational functions, as taught by Goppa, the question of having efficient control on the code distance can be reformulated into a question on the algebraic approximation properties of the corresponding functions. An important global property introduced by Mahler [12] which he called perfectness applies to the functions which are divisible by a polynomial of the type used to model our channel memory. This property is manifested in the polynomial interpolation formula given in terms of Vandermonde determinants. The rational interpolation problem for linear function spaces divisible by $g(z)$ is solvable for arbitrary integers l and m which satisfy $l + m = r$.

5 Conclusion

We have introduced p-adic number and function theory to signal processing. With the aid of this tool we show that it is possible to implement an algebraic decoder that effectively takes advantage of the redundancy introduced by the channel polynomial $g(z)$.

In particular when the roots of $g(z)$ are located at rational division points on the unit circle, its performance can be determined.

References

[1] Hensel, K., "Zahlentheorie", Goschen, Berlin and Leipzig, 1913.

[2] MacDuffee, C.C., "The p-adic numbers of Hensel, American Mathematical Monthly, October 1938, p.500-508.

[3] Kabal, P., Pasupathy, S., "Partial Response Signalling", IEEE Transactions on Communications, Vol.COM-23, No.9, Sept. 1975, pp.921-934.

[4] Cassels, J.W.S., (1986). "Local Fields", London Mathematical Society Student Texts 3, Cambridge University Press, Cambridge.

[5] Krishnamurthy, E.V., Gregory, R.T., "Methods and applications of error-free computation", Springer Verlag, New York, Berlin, Heidelberg, Tokyo, (1984)

[6] Dittenberger, K., "Hensel Codes, an efficient method for exact numerical computation", Diplomarbeit, Johannes Kepler Univ. Linz, 1985.

[7] Alparslan, E., "Finite p-adic computing systems with possible applications", Ph.D. Thesis, University of Maryland, 1975. pp.99-107.

[8] van der Poorten, A., "Hermite Interpolation and p-adic exponential polynomials", Australian Math. Soc. Journal, 22 (1976) 12-26.

[9] Goppa, V.D., "A rational representation of codes and (L,g)-codes", Problems of Information Transmission, Vol. 7, No.3, pp.41-49, July-Sept 1971.

[10] Arndt, H.,"Ein verallgemeinter Kettenbruch-Algorithmus zur rationalen Hermite Interpolation", Numerische Mathematik, vol. 36 (1980), pp.99-107.

[11] Berlekamp, E., "Algebraic Coding Theory", McGraw-Hill Book Company, 1968.

[12] Mahler, K., "Perfect Systems", Compositio Math. (1968), pp.95-166.