IV. EXAMPLES, EVALUATION, AND CONCLUSION

Table I shows the minimum block length necessary to achieve code rates of at least 95%, 90%, and 95% of the (d, k) capacity. This table suggests that even with very simple error-correction schemes, such as single asymmetric or single-shift error correction, one needs a much larger block length than in the no-error correction case, to preserve the code rate. This is somewhat disappointing, and weakens the hypothesis mentioned in Section I: that some of the inherent redundancy of (d, k)-constrained codes can be exploited towards error correction.

Table I

| (d, k) C(d, k) Construction Error Correction Smallest Block Length Necessary to Achieve Percentage of C(d, k) |
|---|---|---|---|---|---|
| (1, 3) 0.5515 | 2 | None | 16 | 1 | 95% |
| (2, 7) 0.5174 | 1 | 1-AS | ≥100 | 5 | 90% |
| (5, 15) 0.3513 | 1 | None | 57 | 28 | 95% |
| | | | 1-AS | 36 | 13 | 90% |
| | | | | 2 | 134 | 62 | 80% |

While the results are not very encouraging considering short codes, it should be pointed out that for long block lengths, it has been demonstrated (see, for example, [14]) that combined coding can offer an improvement over the traditional concatenation of inner (d, k)-constrained codes and outer error-correcting codes.

ACKNOWLEDGMENT

The author would like to thank Patrick Perry for helpful suggestions.

REFERENCES


Combining ECC with Modulation: Performance Comparisons

Mario Blaum

Abstract—A technique for combining ECC with modulation codes of block type is described. Its performance is analyzed with respect to the traditional method in magnetic recording, which involves the concatenation...
tion of an error-correcting code with a convolutional modulation code. Conditions are established under which the new method is superior to the concatenated scheme. In particular, we analyze the performance with respect to the (1,7) and (2,7) modulation constraints. The new construction performs better in a noisy channel dominated by random errors. Cases in which the constraint \( k \) are relaxed is briefly studied.

Index Terms—Decoding, encoding, error-correcting codes, magnetic recording, modulation, modulation codes.

I. INTRODUCTION

A typical encoding configuration for a magnetic or optical recording channel consists of encoding the information bits with an error-correcting code [7] followed by a \((d,k)\) modulation code [9]. The error-correcting code is selected according to the statistics of errors produced by the channel. The choice of the \((d,k)\) constraints for the modulation code depends on the type of signal detection used. The number \( d \) indicates the minimum number of 0's between two consecutive 1's, and its purpose is to reduce intersymbol interference. The number \( k \) indicates the maximum number of 0's between two consecutive 1's. Its choice determines self-clocking properties of the sequences. The modulated sequence is transmitted through a noisy channel and then demodulated. The demodulated sequence goes to a decoder, which attempts to correct possible errors. The output of the decoder is taken as an estimate of the transmitted sequence.

One of the problems with this approach is that the demodulator propagates errors: A single error may become a burst. Hence, the error-correcting code must be a burst-correcting code, even when noise in the channel is dominated by random errors. It should be noted that in general, more redundancy is needed to correct \( t \) bursts than to correct \( t \) random errors.

The most widely used codes for burst correction are Reed–Solomon codes [7]. Reed–Solomon codes are actually byte-correcting codes that are interleaved in order to achieve burst correction. In most applications, the size of a byte is 8 bits, so usually Reed–Solomon codes over \( GF(2^8) \) are considered. By interleaving a byte-coding Reed–Solomon code over \( GF(2^8) \) to depth \( \lambda \), it is possible to correct up to \( t \) bursts of length up to \( \lambda (1-1/2^k) \) bits each.

Popular modulation codes in magnetic recording are the rate 1/2(2,7) [3] and the rate 2/3(1,7) [1] codes. An important characteristic of the (2,7) and the (1,7) codes is that they both have limited error propagation. Specifically, a bit error in the (2,7) code propagates at most over 4 bits after demodulation, while a bit error in the (1,7) code propagates at most over 5 bits. Hence, if the channel produces only random errors, no more than a doubly-interleaved Reed–Solomon code is needed for error correction.

An alternative to the traditional concatenation of ECC and modulation was proposed in [6]. There, the authors use the symbols of a block modulation code in order to construct a single error-correcting code of trellis type. A construction for block modulation codes is given in [10]. Two methods for constructing modulation codes of block type that improve the construction in [10] are presented in [2]. In [6], the authors chose Construction 1 of [2] for their symbols. In this correspondence, we use Construction 2, since, although slightly more complex, it often gives better rates.

The idea is to use a number of block modulated symbols that is a power of 2, say, \( 2^n \). Then, we put these block modulated symbols in one-to-one correspondence with the elements of the finite field \( GF(2^n) \). This way, we can implement Reed–Solomon codes directly over the modulated symbols. This is not a new idea. For instance, it was implemented in the IBM 3480 storage subsystem, where a \((0,3)\) code of rate 8/9 was used [8]. Another example is the Compact Disc, where the modulation code is a \((2,10)\) code of rate 8/16, the so called EFM code [4]. In this correspondence, we compare this block modulation encoding method with the traditional concatenation method and we establish the conditions under which each method is more efficient than the other.

In the next section, we describe the construction in detail. In Section III, we perform the actual comparisons as follows: For a fixed number of information bits, we calculate the total redundancy with the two methods. We then establish conditions under which the redundancy of the new method is smaller than the redundancy of the traditional method. We will conclude that the new method is better for noisy channels strongly dominated by random errors. In particular, we concentrate on the (1,7) and the (2,7) constraints. In Section IV, we relax the conditions on \( k \). We show how we can obtain even better results by allowing \( k \) to increase.

II. BASIC CONSTRUCTION

We start by describing the block modulation construction in [2]. Let \( A_n(d,k,l,r) \) be the set of binary vectors of length \( m \) satisfying the \((d,k)\) constraints and such that each vector has an initial runlength of at most \( l \) 0's and an ending runlength of at most \( r \) 0's. For example, \( A_5(1,7,6,6) \) is given by the following 33 vectors (in lexicographic order):

<table>
<thead>
<tr>
<th>Vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>000001</td>
</tr>
<tr>
<td>000010</td>
</tr>
<tr>
<td>000100</td>
</tr>
<tr>
<td>0000101</td>
</tr>
<tr>
<td>0001000</td>
</tr>
<tr>
<td>0001001</td>
</tr>
<tr>
<td>0001010</td>
</tr>
<tr>
<td>0001011</td>
</tr>
<tr>
<td>0001100</td>
</tr>
<tr>
<td>0001101</td>
</tr>
<tr>
<td>0001110</td>
</tr>
<tr>
<td>0001111</td>
</tr>
<tr>
<td>0010000</td>
</tr>
<tr>
<td>0010001</td>
</tr>
<tr>
<td>0010010</td>
</tr>
<tr>
<td>0010011</td>
</tr>
<tr>
<td>0010100</td>
</tr>
<tr>
<td>0010101</td>
</tr>
<tr>
<td>0010110</td>
</tr>
<tr>
<td>0010111</td>
</tr>
</tbody>
</table>

The symbols of \( A_n(d,k,l,r) \) cannot be used directly for block modulation since we may break the \((d,k)\) constraints when concatenating symbols. For instance, a symbol ending in 1 followed by a symbol starting with 1 would break the \( d \) constraint. Hence, it is necessary to put a certain number of 0's between each symbol. In [2], the authors chose \( l = r = k - d \), and \( d \) the number of merging bits \((2d-1 < k)\). From now on, we only consider the sets \( A_n(d,k,k-d,k-d) \). The precise choice of the merging bits depends on the runlength of 0's at the end of the previous symbol and at the
beginning of the next symbol. A table is given in [2] to that purpose, but it is easy to see what the merging bits should be in the case of \( d = 1 \) and \( d = 2 \).

For instance, let \( s \leq k - d \) be the runlength of \( 0 \)'s at the end of a symbol and \( t \leq k - d \) be the runlength of \( 0 \)'s at the beginning of another symbol. If \( d = 1 \), then the merging bit between the two symbols can be determined to be 0 whenever either \( s = 0 \) or \( t = 0 \), 1 otherwise. In the example above, if symbol \( u_{39} \) is followed by symbol \( u_{40} \), the merging bit is 1, while if symbol \( u_{40} \) is followed by symbol \( u_{39} \), the merging bit is 0. A similar rule may be stated for \( d = 2 \).

Since we are interested in associating symbols in \( A_n(d, k, k - d, k - d) \) with user bytes, we select a subset of \( A_n(d, k, k - d, k - d) \) as big as possible whose cardinality is a power of 2. Let us point out that it is possible to encode all the symbols of \( A_n(d, k, k - d, k - d) \) into an error-correcting code and not merely a subset of it, by using similar techniques to the ones described in [11]. These techniques provide better asymptotic rates than the ones described here, but complexity is also increased.

Denote by \( p(m; d, k) \) a subset of \( A_n(d, k, k - d, k - d) \) whose cardinality is \( 2^v \), where \( v = \lfloor \log_2 |A_n(d, k, k - d, k - d)| \rfloor \) (the symbol \( \lfloor x \rfloor \) denotes the integer part of \( x \)). The set \( p(m; d, k) \) will be our basic block modulation code together with the \( d \) merging bits. Notice that the rate of this block modulation code is \( r = v/(m + d) \). In the last example, \( v = 5 \) and \( r = 5/8 \). Some parameters of block modulation codes with \( r = 8 \) are given in [2].

Let \( f : GF(2^n) \rightarrow p(m; d, k) \) be a one-to-one function. Consider the following construction:

**Construction 1:** Let \( C \) be a \( [k + 2t, k] \) Reed–Solomon code over \( GF(2^n) \), \( n = k + 2t \leq 2^n - 1 \). Then, define the code \( \mathcal{E} \) as follows:

\[
\mathcal{E} = \{(f(u_1), f(u_2), \ldots, f(u_k)) : (u_1, u_2, \ldots, u_k) \in C\}.
\]

Since \( f \) is a one-to-one relationship, the code \( \mathcal{E} \) can correct up to \( t \) symbols of \( p(m; d, k) \) in error. Encoding procedures based on Construction 1 are more or less immediate. We state them in Section 11. In order to make the comparison fair in terms of complexity, we choose a Reed–Solomon code over \( GF(2^n) \), i.e., \( \log_2 |p(m; d, k)| = 2^n \). Let \( r = v/(m + d) \), as shown in the previous section. In general, \( r' < r \), unless we take very large values for \( m \), which may be impractical. As in the previous method, we start with \( \kappa \) information bytes (\( \kappa \) bits). Since we will encode the symbols later into a \( t \)-error-correcting Reed–Solomon code (see Construction 1), we must have \( \kappa + 2t < 2^n \), i.e.,

\[
\kappa < 2^n - 2t.
\]

The redundancy used in encoding the original \( \kappa \) information bits into symbols of \( p(m; d, k) \) is \( \kappa(1/r - 1) \) bits. The \( \kappa \) symbols of \( p(m; d, k) \) are then encoded into a \( t \)-error-correcting Reed–Solomon code over \( GF(2^n) \) as described in Construction 1 with the \( d \) merging bits after each symbol. Hence, we need to add \( 2t \) redundant symbols (\( 2(m + d)t \) redundant bits).
Comparing (1) and (3), some arithmetic manipulations indicated that (1) will be larger than (3) if and only if
\[
\kappa < 2 \left( \frac{1}{\eta - 1} \right) t.
\]
(4)

The region under which the new scheme is better than the old one is determined by inequalities (2) and (4). We unify them in the following lemma.

**Lemma 1:** Given \( \kappa \) information bytes, Construction 1 gives less total redundancy than the usual concatenation scheme if and only if
\[
\kappa < \min \left( 2^\alpha - 2t, 2 \left( \frac{1}{\eta - 1} \right) t \right).
\]
(5)

From Lemma 1, we see that the closer \( r' \) is to \( r \), the bigger the area where the new scheme is more efficient than the old one.

**Example 1:** Consider the rate \( r = 1/2 \) (2,7) code. In this case, (5) becomes
\[
\kappa < \min \left( 2^3 - 2t, 2 \left( \frac{1}{\eta - 1} \right) t \right).
\]
(6)

Table I gives the values of \( r' \) and of \( 2(2r'/(1 - 2r')) - 1 \) for different values of \( \nu \). Using (6) and Table I, we can find the region in which the new construction is better than the old one for different values of \( \nu \). For instance, if \( \nu = 8 \), the new construction has less redundancy whenever \( \kappa < \min(256 - 2t, 30r) \).

**Example 2:** Consider the (1,7) code. In this case, \( r = 2/3 \), hence (5) becomes
\[
\kappa < \min \left( 2^2 - 2t, 2 \left( \frac{3r'}{2 - 3r'} - 1 \right) t \right).
\]
(7)

Table II gives the values of \( r' \) and of \( 2(3r'/(2 - 3r')) - 1 \) for different values of \( \nu \). For instance, the block modulation scheme performs better for \( \nu = 8 \) whenever \( \kappa < \min(256 - 2t, 22t) \). As we can see in Examples 1 and 2, the new scheme is more efficient when the channel is noisy, i.e., the values of \( t \) need to be relatively large.

**IV. Relaxing the Conditions**

Given a \( (d, k) \) modulation code, the \( d \) constraint is by far the more important of the two, since an increase in \( d \) allows greater data density. The \( k \) constraint is not so crucial. If the system allows an increase in \( k \) without complicating too much the process of signal detection, we can obtain even greater improvements with respect to the traditional (1,7) and (2,7) constraints.

Let us start with the (2, \( k \)) constraints. We can verify that \(|A_{1,2}(2,10,8,8)| = 257\), hence, \( \nu = 8 \) and \( r' = 8/13 \). This is the EFM code used in the compact disc (actually, in the compact disc 3 merging bits are used with the EFM code, hence its rate is \( 8/17 \)). So, if we allow \( k \) to increase from 7 to 10, the block modulation scheme will always perform better than the concatenated scheme with the (2,7) code.

If we consider the (1, \( k \)) constraints and \( \nu = 8 \), no gain is obtained by allowing \( k \) to increase to values larger than 7. However, we can verify that \(|A_{1,2}(1,5,4,4)| = 274\), hence, \( r' = 8/13 \) for the (1,5) constraints. This means that a block modulated scheme with the (1,5) constraints performs as well as the same scheme with the (1,7) constraints with respect to the traditional concatenated scheme.

**V. Conclusion**

We have presented the process of error-correcting and modulation in a unified way. The usual way is to concatenate an error-correcting code and a convolutional modulation code. During demodulation and decoding, there is no communication between the two. Here, the idea is to concatenate a Reed–Solomon code as an outer code with a block modulated code as inner code. The inner code is used for error detection only, not for correction.

We concentrated mainly on the (1,7) and the (2,7) modulation constraints, since they are the most widely used in practice. The most important set of symbols are those of cardinality 256, since the industry often uses Reed–Solomon codes over GF(256) whose error-correcting capability does not exceed 8 bytes.

We compared the new scheme with the concatenated scheme for a channel dominated by random errors and peak shifts. We assumed that the Reed–Solomon code in the concatenated scheme is interleaved twice, and it carries a total of \( \kappa \) bytes of information. We gave a condition on \( \kappa \) under which the new scheme performs better than the traditional one. Finally, we studied some situations in which the conditions on the constraint \( k \) are relaxed.

There are some additional advantages in our approach that are not easy to be analytically measured. For example, quite often, a received symbol will not be in our alphabet. That is, errors tend to violate the \( (d, k) \) constraints. This situation will be immediately detected by the decoder, which will declare an erasure at the offending byte. In the traditional scheme, the demodulator does not inform the decoder about forbidden patterns. Since correcting erasures is easier than correcting errors, the error-correcting capability of the code is enhanced.

**Acknowledgment**

The author wishes to express his gratitude to Dr. Kees Immink. The author also thanks the three anonymous referees that made many useful suggestions.

---

**Table I**

VALUES OF \( r' \) AS A FUNCTION OF \( \nu \) FOR THE (2,7) CONSTRAINTS

<table>
<thead>
<tr>
<th>( \nu )</th>
<th>( r' )</th>
<th>( 2(2r'/(1 - 2r')) - 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5/11</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>6/13</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>7/15</td>
<td>26</td>
</tr>
<tr>
<td>8</td>
<td>8/17</td>
<td>30</td>
</tr>
<tr>
<td>9</td>
<td>9/19</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td>10/21</td>
<td>38</td>
</tr>
</tbody>
</table>

**Table II**

VALUES OF \( r' \) AS A FUNCTION OF \( \nu \) FOR THE (1,7) CONSTRAINTS

<table>
<thead>
<tr>
<th>( \nu )</th>
<th>( r' )</th>
<th>( 2(3r'/(2 - 3r')) - 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5/8</td>
<td>28</td>
</tr>
<tr>
<td>6</td>
<td>6/10</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>7/11</td>
<td>40</td>
</tr>
<tr>
<td>8</td>
<td>8/13</td>
<td>22</td>
</tr>
<tr>
<td>9</td>
<td>9/14</td>
<td>52</td>
</tr>
<tr>
<td>10</td>
<td>10/16</td>
<td>28</td>
</tr>
</tbody>
</table>
dc-constrained Error-Correcting Codes with Small Running Digital Sum

Gérard D. Cohen and Simon N. Litsyn

Abstract—Asymptotic bounds on parameters of error-correcting codes with small running digital sum are obtained. The bounds allow us to demonstrate existence of long codes with good error-correcting properties and satisfying some restrictions natural for optical and magnetic recording.

Index Terms—Error correction, dc constraints, combinatorial coding.

I. INTRODUCTION

The problem of constructing line codes for baseband pulse transmission on wire, fiber optic links, and in magnetic or optical recording has attracted significant interest in recent years. Such codes satisfy some natural conditions like constraints on running digital sum (RDS), value of spectra at dc, number of consecutive equal symbols, etc. These restrictions are essential for the parameters of such codes. At the same time the problem of using error correction/detection together with line coding seems to be significant and was considered in some cases of small Hamming distance and methods for combining short (without spectral restrictions) codes to get dc-constrained codes.

The purpose of our paper is to investigate the theoretical restrictions on the parameters of error-correcting codes in the asymptotic case (semi-infinite sequence) when the RDS is upperbounded by some small constant. We do not consider here the minimum Rnul e length as in [1], [2], and [6]. Therefore, as pointed out by one of the referees, our results are more fitted to the optical rather than the magnetic recording channel. For more information, see [13], [15], and [16]. We consider only the binary case but the results of this correspondence can be easily generalized to nonbinary alphabets. All logs are binary, unless explicitly stated.

We use the following notation and definitions. Let us consider a set X of sequences of length n consisting of +1 and -1. Let 
\[ x = (x_1, x_2, \ldots, x_n), \quad x_i \in \{-1, 1\} \]
be such a sequence. We call
\[ \text{RDS}(i, x) = \sum_{j=1}^{i} x_j \]
the running digital sum at time t of sequence x, and denote by
\[ \text{DIS}(X) = \max_{x \in X} \text{RDS}(i, x_1) - \min_{x \in X} \text{RDS}(i, x_2) \]
the imbalance of set X, where max and min are taken over \( t \in \{1, 2, \ldots, n\} \). Without loss of generality we assume the RDS to be centered around 0 (see beginning of Section II). We call sequences from the set X with \( \text{DIS}(X) = m \), m-balanced sequences.

We denote by \( d(x, y) \) the Hamming distance between x and y (number of unequal coordinates), and set
\[ d(X) = \min_{x, y \in X} d(x, y), \quad \frac{|X|}{n} = M(X) = M, \]
\[ R(X) = \log_2 M(X)/n, \quad \delta(X) = d(X)/n, \]
for the minimum distance, size, rate, and relative minimum distance of X, respectively.

Let \( X_n \) denote a maximal code with length n, imbalance at most m and relative minimum distance at least \( \delta \). The rate of \( X_n \) is a function \( R(n, m, \delta) \). We shall deal with the behavior of this quantity for all large enough n and set
\[ R_m(\delta) = R(n, m, \delta). \]

Let us denote a code with length n, number of words M, minimum distance d and imbalance m by \( (n, M, d, m) \)-code. We write \( (n, M, d) \) when m is irrelevant.

For the description of sequences we may also use the sequence of RDS at consecutive moments. Hence, if we consider a sequence \( x = (x_1, x_2, \ldots, x_n) \in X \), we use vector
\[ r = (r_1, r_2, \ldots, r_n), \quad r_i \in [\min \text{RDS}, \max \text{RDS}], \quad r_{j+1} - r_j = x_{j+1}, \quad r_0 = 0, \quad j = 0, 1, \ldots, n-1. \]

Evidently, \( |r_{j+1} - r_j| = 1 \) and binary sequences can have only odd RDS at odd moments and even RDS at even moments.

The following proposition shows that the constraints on imbalance also limit the number of consecutive equal symbols.

Proposition 1: Let \( \text{DIS}(X) = m \). Then the maximal number of consecutive equal symbols does not exceed m.

REFERENCES

[12] G. D. Cohen is with the ENST, 46 rue Barrault, 75634 Paris cedex 13, France.