# DC-Free Binary Convolutional Coding

Tadashi Wadayama, *Member, IEEE,* and A. J. Han Vinck, *Senior Member, IEEE*

***Abstract*—A novel DC-free binary convolutional coding scheme is presented. The proposed scheme achieves the DC-free coding and error-correcting capability simultaneously. The scheme has a simple cascaded structure of the running digital sum (RDS) control encoder and the conventional convolutional encoder. A given sequence becomes DC-free if and only if the absolute RDS value of the sequence is bounded by a constant for any time instant. The RDS control encoder generates a sequence which gives the convolutional-coded sequence with a bounded RDS value. The structure allows us to exploit efficient soft-decision decoding which attains additional coding gains compared with hard-decision decoding over an additive white Gaussian noise (AWGN) channel. Bounds on the RDS value are explicitly established for the proposed scheme. By using the bounds, we have performed computer searches for finding good RDS control encoders. The proposed scheme provides wide varieties of reasonable tradeoffs between the coding gain, the RDS constraint, and decoding complexity. For example, a 64-state DC-free coding scheme with the overall rate $6/16$ and the minimum free distance $10$ has been obtained. This scheme satisfies a bounded RDS constraint (from $-18$ to $+18$) and it yields a considerably high asymptotic coding gain (over an AWGN channel) of 5.7 dB.**

***Index Terms*—Additive encoding, convolutional code, DC-free coding.**

## I. Introduction

THE DC-free coding is widely employed in digital communication and storage areas. “DC-free” means that the coded sequence has no DC spectral component. A DC-free or DC-suppressed coding is essential in some baseband transmission and magnetic/optical recording systems.

For a noiseless channel, many DC-free codes have been devised and the details about them can be found in [1]. A number of intensive researches on DC-free codes with an additional error-correcting capability [2]–[6] have been reported as well. For bandwidth-limited channels, trellis coding techniques with the DC-free property have been investigated in [7]–[9]. By using such a code, we can obtain not only the DC-freeness but also additional coding gains over a noisy channel.

It is natural to consider the combination of DC-free coding and binary convolutional coding for error control along the line of the above mentioned research. Binary convolutional codes are, in particular, matched to power-limited channels and they are commonly utilized in practical communication systems as a crucial part of an error control system. One of the advantages of the combination is that it enables us to exploit efficient soft-decision decoding algorithms such as the Viterbi algorithm. In general, soft-decision decoding gives additional coding gains compared with hard-decision decoding. However, the combination is not so straightforward from the earlier works. The trellis coding techniques [7]–[9] are based on the signal constellation expansion. Thus, we cannot directly apply the ideas to the power-limited channel cases. On the other hand, most of the known DC-free block codes with the error correcting capability are based on nonlinear codes such as balanced codes. Those codes may not have a simple trellis structure which supports the efficient soft-decision decoding. In [2], a DC-free code generated by a finite-state machine with coding rate $2/4$ and minimum free distance $4$ is presented. Although the code itself has excellent properties, a generalization of the code construction to other code parameters seems to be difficult.

There are several works on DC-free error-correcting codes based on convolutional codes. Deng, Li, and Herro [10] presented a DC-free error-correcting convolutional coding technique. In their method, the all–1’s vector in the generator matrix of a convolutional code is exploited to control the running digital sum (RDS) of encoded sequences. Nasiri-Kenari and Rushforth [11] investigated DC-free subcodes of convolutional codes. Recently, Chiu [12] showed DC-free error-correcting codes based on convolutional codes. In Chiu’s scheme, a codeword of a convolutional codes with a small RDS value is chosen with the Viterbi algorithm. These methods seem promising and further investigation on binary DC-free coding schemes with a simple trellis structure, or equivalently, with small decoding complexity is hoped for.

In this paper, we present a novel DC-free convolutional coding scheme with an error correcting capability. Fig. 1 represents the architecture of our proposed scheme. First, the user message sequence $(u_0u_1u_2\cdots)$ is encoded to the *intermediate sequence* $(x_0x_1x_2\cdots)$ by an *RDS control encoder*. The convolutional encoder then converts an intermediate sequence to the coded sequence $(y_0y_1y_2\cdots)$. After the ordinary binary-bipolar conversion, the coded sequence is transmitted over a noisy channel such as the additive white Gaussian noise (AWGN) channel.

The term “RDS” means the running digital sum of a (bipolar) coded sequence. It is well known that the DC-free property is achieved if and only if the absolute value of the RDS is bounded by a constant value for any time instant [1]. The RDS control encoder must generate an intermediate sequence which gives a coded sequence with a desired RDS constraint. In other words, an intermediate sequence should be determined in such a way that it generates coded sequences with a bounded RDS.

Manuscript received May 9, 2000; revised June 14, 2001. The material in this paper was presented at the IEEE International Symposium on Information Theory, Sorrento, Italy, June 2000.

T. Wadayama is with the Faculty of Computer Science and System Engineering, Okayama Prefectural University, Okayama, 719-1197, Japan (e-mail: wadayama@c.oka-pu.ac.jp).

A. J. H. Vinck is with the Institute for Experimental Mathematics, Essen University, 45326 Essen, Germany (e-mail: vinck@exp-math.uni-essen.de).

Communicated by E. Soljanin, Associate Editor for Coding Techniques.

Publisher Item Identifier S 0018-9448(02)00055-X.

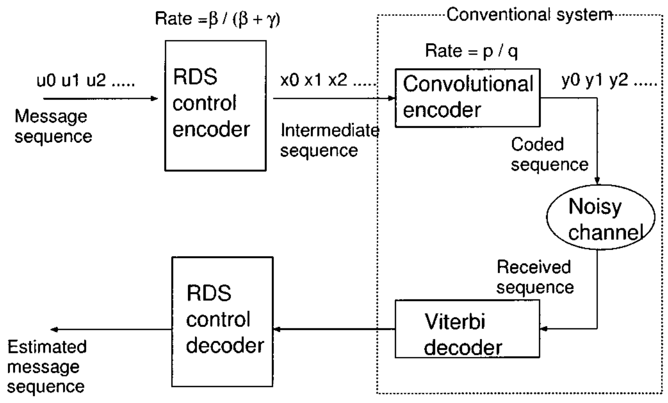


Fig. 1. Architecture of DC-free convolutional coding scheme.

Fig. 1 presents the cascaded structure of the RDS control encoder/decoder and the convolutional encoder/Viterbi decoder. With this architecture, we are able to exploit soft-decision decoding with the Viterbi algorithm. Moreover, the dashed box part in Fig. 1 is exactly identical to a conventional convolutional coding system. The RDS control encoder and decoder can be regarded as a front-end and a back-end of the conventional convolutional coding system. Thus, we can use a ready-made CODEC to implement the proposed scheme.

The proposed scheme is based on the following three major ideas: 1) *additive encoding* using a binary linear block code, 2) upper and lower bounds on the RDS for an additive encoder, and 3) splitting a convolutional code into infinite sequences of a linear block code, which is called a *window code*. In the following, we shall explain these ideas in order.

The organization of the paper is as follows. In Section II, we first develop a DC-free coding scheme with an additive encoder. We next prove basic properties of an additive encoder including new upper and lower bounds on the RDS. Section III includes the principal results of this paper such as the details on the DC-free convolutional coding scheme and its performance. Good RDS control encoders which have been found by computer search and their performance are also shown. In Section IV, we give a summary.

## II. DC-Free Coding Scheme Based on an Additive Encoder

In this section, first, we will introduce necessary notations and definitions Then, a DC-free coding scheme based on an additive encoder is presented. The scheme can be considered as a modified version of the scheme presented by Deng and Herro [5] and has a close relationship to the idea of additive coding [13]. The class of additive encoders presented here can also be regarded as a subclass of guided scrambling [14], or as a multimode code [15].

### A. Notation and Definition

Let $\{s_0, s_1, \ldots, s_t, \ldots\}$ be an semi-infinite length bipolar $\{+1, -1\}$-valued sequence. The RDS of the sequence is defined by $z_t \triangleq \sum_{j=0}^{t} s_j$. If the RDS of the sequence is bounded by a constant $c > 0$ for any time instant $t$ such that $|z_t| \le c$, then the sequence has a spectral null at DC [1].

For $\boldsymbol{v} = (v_0, v_1, \ldots, v_{n-1}) \in \{0, 1\}^n$, we define the vector RDS of $\boldsymbol{v}$ by

$$S(\boldsymbol{v}) \triangleq \sum_{j=0}^{n-1} f(v_j).$$

The binary-bipolar conversion mapping $f$ is defined by

$$f(l) \triangleq \begin{cases} -1, & l = 0 \\ +1, & l = 1. \end{cases}$$

The upper and lower RDS of $\boldsymbol{v}$ are defined by

$$U(\boldsymbol{v}) \triangleq \max_{0 \le t \le n-1} \sum_{j=0}^{t} f(v_j) \qquad L(\boldsymbol{v}) \triangleq \min_{0 \le t \le n-1} \sum_{j=0}^{t} f(v_j).$$

Let $C$ be an $(n, k, d)$ binary linear block code, where $n$, $k$, and $d$ denote the length, the dimension, and the minimum distance, respectively. Consider the following decomposition $(C_0, C_1)$ of $C$.

*Definition 1 (Direct Sum Decomposition):* For a given binary linear code $C$, if two binary linear codes $C_0$ and $C_1$ satisfy

$$C = \{\boldsymbol{c}_0 \oplus \boldsymbol{c}_1 : \boldsymbol{c}_0 \in C_0, \boldsymbol{c}_1 \in C_1\}$$

and

$$C_0 \cap C_1 = \mathbf{0}$$

then we call the pair of codes $(C_0, C_1)$ the *direct sum decomposition* of $C$. The code $C$ is called the direct sum code based on $C_0$ and $C_1$. □

Let $k_0$ and $k_1$ be the dimensions of $C_0$ and $C_1$, and $G_0$ and $G_1$ be the generator matrices of $C_0$ and $C_1$, respectively. From the definition above, it is obvious that the equality $k = k_0 + k_1$ holds. We assume two one-to-one mappings called encoding mappings

$$\psi_0: F_2^{k_0} \to C_0$$
$$\psi_1: F_2^{k_1} \to C_1$$

where $F_2$ is the Galois field with two elements $\{0, 1\}$. We denote the addition over $F_2$ by $\oplus$.

### B. Additive Encoder

We here give the definition of an additive encoder and a selection rule used in an additive encoder.

Assume an infinite-length binary message sequence $\{\boldsymbol{a}_0, \boldsymbol{a}_1, \ldots\}$. Each vector $\boldsymbol{a}_i (i = 0, 1, 2, \ldots)$ belongs to $F_2^{k_1}$. An additive encoder encodes a message block $\boldsymbol{a}_i$ to $\boldsymbol{c}_i \in C$ for each block index $i$. The code $C$ is a binary linear code of length $n$. The resulting sequence $\{\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots\}$ is called a coded sequence. The additive encoder appends redundancy $k_0 = k - k_1$ bits per block and thus the coding rate becomes $k_1/n$. After the binary-bipolar conversion, the bipolar sequence $\{f(\boldsymbol{c}_0), f(\boldsymbol{c}_1), \ldots\}$ is transmitted over the noisy channel.

*1) Definition:* For achieving DC-free transmission, the additive encoder has to generate the coded sequence with an RDS constraint. The following definition of an additive encoder is quite simple.

*Definition 2 (Additive Encoder):* An additive encoder encodes a message block $\boldsymbol{a}_i$ into $\boldsymbol{c}_i$ in the following way:

$$\boldsymbol{c}_i = \psi_0(\boldsymbol{b}_i) \oplus \psi_1(\boldsymbol{a}_i)$$

where $\boldsymbol{b}_i \in F_2^{k_0}$ is selected by the additive encoder according to the value of the RDS $\sum_{j=0}^{i-1} S(\boldsymbol{c}_j)$ and a *selection rule*. □

We call the vector $\boldsymbol{b}_i$ the *control vector*. In other words, the additive encoder has freedom to select a control vector and should specify a control vector so as to obtain a code sequence which keeps the RDS value bounded.

Note that the codeword $\boldsymbol{c}_i$ belongs to $C$ for any block index $i$ because $C$ is the direct sum code of $C_0$ and $C_1$. It guarantees $d_h(\boldsymbol{c}_i, \boldsymbol{c}_i') \geq d$ for any $\boldsymbol{c}_i \neq \boldsymbol{c}_i'$, where $d_h(\cdot, \cdot)$ is the Hamming distance between two vectors. That is, the coded sequences produced by an additive encoder have error correcting capability inherited from the direct sum code $C$.

The work by Deng and Herro [5] is the first dealing with this class of encoders. In [5], they impose relatively strong restrictions on the decomposition of $C$ and use another selection rule. Here, we have removed such restrictions because we need to treat any decomposition of $C$. This is the main difference between their approach and ours.

The RDS constraint achieved by an additive encoder depends on the decomposition of $C$. For a given $\boldsymbol{c}_1 \in C_1$, we define the set $V(\boldsymbol{c}_1)$ by

$$V(\boldsymbol{c}_1) = \{\boldsymbol{c}_0 \oplus \boldsymbol{c}_1 \colon \boldsymbol{c}_0 \in C_0\}.$$

The set $V(\boldsymbol{c}_1)$ is called the *shell* of $\boldsymbol{c}_1$. Namely, the shell of $\boldsymbol{c}_1$ is a coset of $C_0$ containing $\boldsymbol{c}_1$. If for any $\boldsymbol{x} \in C_1$ there exist $\boldsymbol{v}_1, \boldsymbol{v}_2 \in V(\boldsymbol{x})$ satisfying $S(\boldsymbol{v}_1) \geq 0$ and $S(\boldsymbol{v}_2) \leq 0$, then the decomposition $(C_0, C_1)$ is called a *good decomposition*. Otherwise, the decomposition is called a *bad decomposition*.

*2) Selection Rule:* The RDS constraint property of an additive encoder also heavily depends on the selection rule. A good selection rule is required to reduce the absolute value of the RDS as much as possible. Smaller absolute values of the RDS are preferable with respect to near-DC spectral component suppression.

We introduce a selection rule for an additive encoder. The rule is used for specifying the control vectors. The following selection rule includes the values of $U(\cdot)$ and $L(\cdot)$. We show later the RDS bound for the selection rule depends on these values.

*Definition 3 (Selection Rule):* For every $\boldsymbol{x} \in C_1$, the vector $\boldsymbol{x}^+$ is chosen such that

$$\boldsymbol{x}^+ \in \{\boldsymbol{v} \in V(\boldsymbol{x}) \colon S(\boldsymbol{v}) \geq 0\}$$

and $\boldsymbol{x}^+$ has to satisfy the following additional inequality:

$$S(\boldsymbol{x}^+) - L(\boldsymbol{x}^+) \leq S(\boldsymbol{v}') - L(\boldsymbol{v}') \tag{1}$$

for any $\boldsymbol{v}' \in V(\boldsymbol{x})$.[1] If there are several vectors that satisfy the above conditions, the vector $\boldsymbol{v}$ which gives the smallest $U(\boldsymbol{v})$ should be chosen as $\boldsymbol{x}^+$. For a given $\boldsymbol{x}$, $\boldsymbol{x}^+$ should be uniquely defined. In a similar way, $\boldsymbol{x}^-$ is chosen in such a way

$$\boldsymbol{x}^- \in \{\boldsymbol{v} \in V(\boldsymbol{x}) \colon S(\boldsymbol{v}) \leq 0\}$$

and $\boldsymbol{x}^-$ has to satisfy the following additional inequality:

$$S(\boldsymbol{x}^-) - U(\boldsymbol{x}^-) \geq S(\boldsymbol{v}') - U(\boldsymbol{v}') \tag{2}$$

for any $\boldsymbol{v}' \in V(\boldsymbol{x})$. If there are several vectors satisfying the above conditions, the vector $\boldsymbol{v}$ which gives the largest $L(\boldsymbol{v})$ should be chosen as $\boldsymbol{x}^-$. For a given $\boldsymbol{x}$, $\boldsymbol{x}^-$ should be uniquely defined. Assume that the message block $\boldsymbol{a}_i$ is given and $\boldsymbol{x} = \psi_1(\boldsymbol{a}_i)$ holds. The additive encoder selects the control vector $\boldsymbol{b}_i$ satisfying

$$\psi_0(\boldsymbol{b}_i) \oplus \psi_1(\boldsymbol{a}_i) = \begin{cases} \boldsymbol{x}^+, & \text{if } \sum\limits_{j=0}^{i-1} S(\boldsymbol{c}_j) \leq 0, \\ \boldsymbol{x}^-, & \text{if } \sum\limits_{j=0}^{i-1} S(\boldsymbol{c}_j) > 0. \end{cases} \tag{3}$$

□

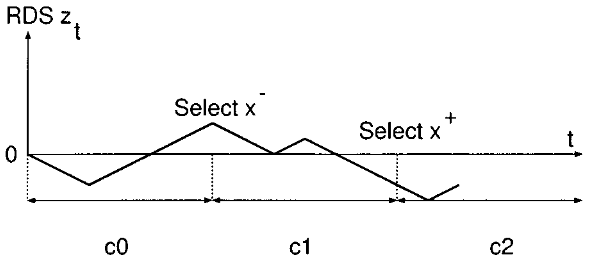


Fig. 2. Typical RDS as a function of time in the control vector selection procedure.

An additive encoder keeps the RDS value for every time instant in its memory and uses this value to select a new control vector. Fig. 2 explains the selection procedure. After processing $\boldsymbol{c}_0$, we have a positive RDS value. At this moment, the additive encoder selects $\boldsymbol{b}_1$ corresponding to $\boldsymbol{x}^-$. From the definition of $\boldsymbol{x}^-$, it has a nonpositive vector RDS value. As a result, the RDS value is decreasing in a time interval corresponding to $\boldsymbol{c}_1$. Obviously, the DC-free condition (i.e., the RDS is bounded at any time instant) holds if $(C_0, C_1)$ is a good decomposition.

## C. Upper and Lower Bounds on RDS

In order to construct a good additive encoder, we need tools for performance evaluation of an additive encoder. We here to prove upper and lower bounds on the RDS of the coded sequences generated by an additive encoder with the selection rule defined above. The bound will also play a key role in the design of the DC-free convolutional coding to be discussed later.

We first discuss a bound for the RDS $z_t$ in the case where $t$ is a multiple of block length $n$, namely, $t = ni (i \geq 0)$.

*Lemma 1:* For $i \geq 0$

$$\min\left\{S(\boldsymbol{x}^-) \colon \boldsymbol{x} \in C_1\right\} \leq z_{ni} \leq \max\left\{S(\boldsymbol{x}^+) \colon \boldsymbol{x} \in C_1\right\}. \tag{4}$$

*Proof:* The initial condition $z_0 = 0$ holds for $i = 0$. We here assume that the claim of the lemma holds for the block index $i-1$. If $\min\{S(\boldsymbol{x}^-) \colon \boldsymbol{x} \in C_1\} \leq z_{n(i-1)} \leq 0$, then $\boldsymbol{x}^+$ is

[1]We use the notation $\boldsymbol{x}^+$ to show the dependency on $\boldsymbol{x}$. To clarify the dependency, a function form notation like $\phi_+(\boldsymbol{x})$ might be better but we think the function form is somewhat cumbersome. Thus, this unconventional notation is used throughout the paper.

chosen as the next codeword according to Rule-A or to Rule-B. In this case, we have the following inequalities:

$$z_{ni} = z_{n(i-1)} + S(\boldsymbol{x}^+) \le z_{n(i-1)} + \max\left\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\} \le \max\left\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\}$$

and

$$z_{ni} = z_{n(i-1)} + S(\boldsymbol{x}^+) \ge z_{n(i-1)} \ge \min\left\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\}.$$

On the other hand, if $0 < z_{n(i-1)} \le \max\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\}$ holds, then $\boldsymbol{x}^-$ is chosen as the next codeword. In this case, we have

$$z_{ni} = z_{n(i-1)} + S(\boldsymbol{x}^-) \le z_{n(i-1)} \le \max\left\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\}$$

and

$$z_{ni} = z_{n(i-1)} + S(\boldsymbol{x}^-) \ge z_{n(i-1)} + \min\left\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\} \ge \min\left\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\}.$$

By induction, we can complete the proof of the claim of the lemma. □

The lemma naturally leads to the following theorem on the RDS.

*Theorem 1 (RDS Bound for an Additive Encoder):* For any time instant $t \ge 0$, the following inequalities hold:

$$z_t \le \max\{\max\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\} + \max\{U(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\}, \max\{U(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\}\} \triangleq \mathcal{U}(C_0, C_1)$$

$$z_t \ge \min\{\min\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\} + \min\{L(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\}, \min\{L(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\}\} \triangleq \mathcal{L}(C_0, C_1).$$

*Proof:* From the previous lemma, we can assume that

$$\min\left\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\} \le z_{ni} \le \max\left\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\}$$

for $i \ge 0$. We now consider the values $z_{ni+j}$ for $0 \le j \le n-1$. If $\min\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\} \le z_{ni} \le 0$, then $\boldsymbol{x}^+$ is chosen as the next codeword. For $0 \le j \le n-1$, we have

$$z_{ni+j} = z_{ni} + \sum_{l=0}^{j} f(x_l^+) \le z_{ni} + \max\left\{U(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\} \le \max\left\{U(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\}$$

and

$$z_{ni+j} = z_{ni} + \sum_{l=0}^{j} f(x_l^+) \ge z_{ni} + \min\left\{L(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\} \ge \min\left\{S(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\} + \min\left\{L(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\}.$$

If $0 < z_{ni} \le \max\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\}$, then $\boldsymbol{x}^-$ is chosen as the next codeword. For $0 \le j \le n-1$, we have

$$z_{ni+j} = z_{ni} + \sum_{l=0}^{j} f(x_l^-) \le z_{ni} + \max\left\{U(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\} \le \max\left\{S(\boldsymbol{x}^+)\colon \boldsymbol{x} \in C_1\right\} + \max\left\{U(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\}$$

and

$$z_{ni+j} = z_{ni} + \sum_{l=0}^{j} f(x_l^-) \ge z_{ni} + \min\left\{L(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\} \ge \min\left\{L(\boldsymbol{x}^-)\colon \boldsymbol{x} \in C_1\right\}.$$

Combining these inequalities on $z_{ni+j}$, we obtain the claim of the theorem. □

TABLE I
VECTOR RDS, UPPER, LOWER VALUES

| $\boldsymbol{x}$ | $\boldsymbol{y}$ | $S(\boldsymbol{y})$ | $U(\boldsymbol{y})$ | $L(\boldsymbol{y})$ | Rule-A | Rule-B |
|---|---|---|---|---|---|---|
| 0000 | 0000 | −4 | 0 | −4 | | |
| | 1100 | 0 | 2 | 0 | $x^-$ | $x^+$ |
| | 0011 | 0 | 0 | −2 | $x^+$ | $x^-$ |
| | 1111 | 4 | 4 | 0 | | |
| 1000 | 1000 | −2 | 1 | −2 | $x^-$ | |
| | 0100 | −2 | 0 | −2 | | $x^-$ |
| | 1011 | 2 | 2 | 0 | | $x^+$ |
| | 0111 | 2 | 2 | −1 | $x^+$ | |
| 0010 | 0010 | −2 | 0 | −2 | | $x^-$ |
| | 1110 | 2 | 3 | 0 | | |
| | 0001 | −2 | 0 | −3 | $x^-$ | |
| | 1101 | 2 | 2 | 0 | $x^+$ | $x^+$ |
| 1010 | 1010 | 0 | 1 | 0 | | $x^+$ |
| | 0110 | 0 | 1 | −1 | $x^+$ | |
| | 1001 | 0 | 1 | −1 | $x^-$ | |
| | 0101 | 0 | 0 | −1 | | $x^-$ |

TABLE II
VALUES RELATED TO THE RDS BOUND

| | Rule-A | Rule-B |
|---|---|---|
| $\max\{S(\boldsymbol{x}^+) : \boldsymbol{x} \in C_1\}$ | 2 | 2 |
| $\max\{U(\boldsymbol{x}^-) : \boldsymbol{x} \in C_1\}$ | 2 | 0 |
| $\min\{S(\boldsymbol{x}^-) : \boldsymbol{x} \in C_1\}$ | −2 | −2 |
| $\min\{L(\boldsymbol{x}^+) : \boldsymbol{x} \in C_1\}$ | −2 | 0 |
| $\max\{U(\boldsymbol{x}^+) : \boldsymbol{x} \in C_1\}\}$ | 2 | 2 |
| $\min\{L(\boldsymbol{x}^-) : \boldsymbol{x} \in C_1\}\}$ | −3 | −2 |

The bound can be evaluated for any combination of $C$ and its decomposition. The time complexity for computing the bound is $O(|C|)$.

*Example 1:* Let

$$G_0 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \qquad G_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{5}$$

The decomposition is a good decomposition for $C = F_2^4$. Several parameters including the vector RDS are listed in Table I. Consider a simplified selection rule, which is called Rule-A. Rule-A is almost the same as the selection rule defined in Section II-B2 (called Rule-B here). The only difference is that Rule-A does not include inequalities (1) and (2). It is easy to show the bounds on RDS presented in this section are also valid for Rule-A. The vectors $\boldsymbol{x}^+$ and $\boldsymbol{x}^-$ according to Rule-A and Rule-B for each codeword of $C_1$ are also shown in Table I. From Table I, we can obtain all the values which are needed to compute the bounds in Theorem 1. These values are presented in Table II. From Table II and Theorem 1, we have $-4 \le z_t \le 4$ for Rule-A. On the other hand, a tighter RDS constraint, $-2 \le z_t \le 2$, can be obtained with Rule-B. This result shows the superiority of Rule-B over Rule-A. The problem of Rule-A is that it does not care about the values $U(\cdot)$ and $L(\cdot)$. The bound in Theorem 1 depends not only on the vector RDS $S(\cdot)$ but also considerably on the values $U(\cdot)$ and $L(\cdot)$. We have designed Rule-B taking these values into account.

We do not claim that Rule-B is the optimum selection rule in terms of the RDS constraint. However, some experiments indicate that Rule-B gives relatively tighter upper and lower bounds in Theorem 1 than other rules. At least, in most cases, Rule-B is much superior to Rule-A. Therefore, we only use Rule-B as the selection rule in this paper. □

*Example 2:* The following generator matrices give a direct sum decomposition of the $(8, 4, 4)$ extended Bose–Chaudhuri–Hocquenghem (BCH) code:

$$G_0 = (1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1)$$

$$G_1 = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

From Theorem 1, we have the bound $-10 \le z_t \le 10$.

Next, we shall examine another decomposition of the extended BCH code. Let

$$G_0 = (0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1)$$

$$G_1 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

From Theorem 1, we have the bound $-4 \le z_t \le 4$. This decomposition is superior to the former decomposition in terms of the RDS constraint. □

The above example explains that the choice of a direct sum decomposition of $C$ is crucial for achieving a tight RDS constraint.

*Example 3:* Let

$$G_0 = \begin{pmatrix} 0101010101010101 \\ 0011001100110011 \\ 0000000011111111 \end{pmatrix}$$

$$G_1 = \begin{pmatrix} 0001000100010001 \\ 0000100100000110 \\ 0000010100000101 \\ 0000001100000011 \\ 1000000100010111 \\ 0000000001010101 \\ 0000000000110011 \\ 0000000000001111 \end{pmatrix}.$$

These generator matrices correspond to a direct sum decomposition of the $(16, 11, 4)$ extended BCH code. From Theorem 1, we have the bound $-6 \le z_t \le 6$. This coding scheme has at least the error correcting capability of the $(16, 11, 4)$ extended BCH code. It is interesting to compare this coding scheme with other known DC-free coding schemes (see Table III). From Table III, we conclude that the presented coding scheme is almost comparable to the Deng–Herro scheme and slightly inferior to the Ferreira and the Blaum scheme. □

## III. DC-Free Convolutional Coding

In this section, we present a DC-free convolutional coding scheme. The main idea is to apply the additive encoder idea to window codes obtained from a convolutional code. We can obtain a window code by splitting a convolutional code into an infinite series of block codes.

TABLE III
BLOCK DC-FREE CODES WITH ERROR CORRECTING CAPABILITY ($n = 15, 16$)

| $n$ | $k'$ | $d$ | $c$ | Reference |
|---|---|---|---|---|
| 16 | 8 | 4 | 5 | Ferreira[3] |
| 16 | 9 | 4 | 5 | Blaum[4] |
| 15 | 8 | 4 | 7 | Deng and Herro[5] |
| 16 | 8 | 4 | 6 | Example 3 |

$n$: code length, $k'$:information bits contained in a codeword,
$d$: minimum Hamming distance, $c$: upper bound on $|z_t|$.

### A. Splitting a Convolutional Code Into Window Codes

Here, the notation concerned with a convolutional code is briefly introduced. Then, the definition of the window code is given.

*1) Notation on Convolutional Codes:* Let a binary input sequence $\boldsymbol{x}_0\boldsymbol{x}_1\boldsymbol{x}_2\cdots$ of infinite length be encoded by a convolutional encoder, where $\boldsymbol{x}_i$ is a binary $p$-tuple.

A convolutional encoder is defined as follows. Let $g_i(i = 0, 1, 2, \ldots, m)$ be a $p \times q$ matrix over $F_2$. The parameter $m$ is called the *encoder memory*. We refer to the matrices as the *generator submatrices*. The convolutional encoder encodes the input sequence according to the following rule:

$$\boldsymbol{y}_i = \boldsymbol{x}_i g_0 \oplus \boldsymbol{x}_{i-1} g_1 \oplus \cdots \oplus \boldsymbol{x}_{i-m} g_m, \qquad i = 0, 1, 2, \ldots \quad (6)$$

and outputs $\boldsymbol{y}_i (i = 0, 1, 2 \ldots)$ as the code sequence, where $\boldsymbol{y}_i$ is a binary $q$-tuple. We assume $\boldsymbol{x}_i = \mathbf{0}$ for $i < 0$ for formality. Hereinafter, we call the set of all the allowable code sequences defined by (6) the *convolutional code* $C$.

We can also write the encoding rule of the convolutional encoder as $(\boldsymbol{y}_0\boldsymbol{y}_1\boldsymbol{y}_2\cdots) = (\boldsymbol{x}_0\boldsymbol{x}_1\boldsymbol{x}_2\cdots)G$, where the *generator matrix* $G$ is the infinite size matrix represented by

$$G = \begin{pmatrix} g_0 & g_1 & \cdots & g_m & & & \\ & g_0 & g_1 & \cdots & g_m & & \\ & & g_0 & g_1 & \cdots & g_m & \\ & & & \vdots & \cdots & & \vdots \end{pmatrix}. \quad (7)$$

*2) Definition of the Window Code:* We define the *window matrix* of $G$ as follows.

*Definition 4 (Window Code):* For a given $\alpha \in \{0, 1, \ldots\}$, the window matrix of $G$ is the $\ell \times r$ submatrix of $G$ which has the form

$$G^\alpha = \begin{pmatrix} g_m & & & & & & & & \\ g_{m-1} & g_m & & & & & & & \\ g_{m-2} & g_{m-1} & g_m & & & & & & \\ \vdots & \vdots & \vdots & & & & & & \\ g_0 & g_1 & g_2 & \cdots & g_m & & & & \\ & g_0 & g_1 & g_2 & \cdots & g_m & & & \\ & & g_0 & g_1 & g_2 & \cdots & g_m & & \\ & & & \vdots & \vdots & \vdots & \vdots & & \\ & & & & & g_0 & g_1 & g_2 \\ & & & & & & g_0 & g_1 \\ & & & & & & & g_0 \end{pmatrix} \quad (8)$$

where $\ell \triangleq (2m + 1 + \alpha)p$ and $r \triangleq (m + 1 + \alpha)q$. The binary linear block code generated by $G^\alpha$ is called the *window code* of $C$, which is denoted by $C^\alpha$. □

*Example 4:* We here consider a binary four-state convolutional code of rate $1/2$ defined by the generator submatrices

$$g_0 = (11) \qquad g_1 = (10) \qquad g_2 = (11).$$

In this case, the generator matrix $G$ has the form

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 1 & & & & \\ & & 1 & 1 & 1 & 0 & 1 & 1 & & \\ & & & & 1 & 1 & 1 & 0 & 1 & 1 \\ & & & & & & \vdots & \vdots & \vdots & \end{pmatrix}.$$

For $\alpha = 0$, we have the $5 \times 6$ window matrix such that

$$G^0 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \tag{9}$$

When $\alpha = 1$, we have the $6 \times 8$ window matrix

$$G^1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \tag{10}$$

□

From the encoding rule in (6) and the definition of the window matrix, we can obtain another expression of the encoding rule based on the window matrix such that

$$(\boldsymbol{y}_t \boldsymbol{y}_{t+1} \cdots \boldsymbol{y}_{t+m+\alpha}) = (\boldsymbol{x}_{t-m} \boldsymbol{x}_{t-m+1} \cdots \boldsymbol{x}_{t+m+\alpha}) G^\alpha \tag{11}$$

for $t \geq 0$.

*3) Direct Sum Decomposition of the Window Code:* In order to exploit the results on the additive encoder, a direct sum decomposition of $C^\alpha$ is needed. We here discuss a direct sum decomposition of $C^\alpha$.

Let $M$ be a $(\gamma+\beta)\times(\gamma+\beta)$-binary nonsingular matrix, where $\gamma$ and $\beta$ are positive integers satisfying $\ell = pm + \gamma + \beta$. The matrix $M$ is called a *decomposition matrix*. We now consider the two submatrices of a decomposition matrix $M$ which are denoted by $M_0$ and $M_1$. The matrix $M_0$ is the $\gamma \times (\gamma+\beta)$-matrix which consists of the first $\gamma$-rows of $M$. The matrix $M_1$ is the $\beta \times (\gamma + \beta)$-matrix which consists of the last $\beta$-rows of $M$. Thus, we have the decomposition of $M$ such that

$$M = \begin{pmatrix} M_0 \\ M_1 \end{pmatrix}. \tag{12}$$

For a given window code $C^\alpha$ and a decomposition matrix $M$, two subcodes of $C^\alpha$ are defined by

$$C_0^\alpha \triangleq \{(\mathbf{0}^{pm} | \boldsymbol{u} M_0)\, G^\alpha \colon \boldsymbol{u} \in F_2^\gamma\} \tag{13}$$

$$C_1^\alpha \triangleq \left\{(\boldsymbol{v} | \boldsymbol{w} M_1)\, G^\alpha \colon \boldsymbol{v} \in F_2^{pm},\, \boldsymbol{w} \in F_2^\beta\right\}. \tag{14}$$

The operator | is the concatenation operator of two vectors and $\mathbf{0}^n$ means the zero vector of length $n$. For any $\boldsymbol{a} \in C_0^\alpha$ and $\boldsymbol{b} \in C_1^\alpha$, we have

$$\begin{aligned} \boldsymbol{a} \oplus \boldsymbol{b} &= (\mathbf{0}^{pm} | \boldsymbol{u} M_0) G^\alpha \oplus (\boldsymbol{v} | \boldsymbol{w} M_1) G^\alpha \\ &= (\boldsymbol{v} | \boldsymbol{u} M_0 \oplus \boldsymbol{w} M_1) G^\alpha \\ &= (\boldsymbol{v} | (\boldsymbol{u} | \boldsymbol{w}) M) G^\alpha \\ &= (\boldsymbol{v} | \boldsymbol{v}') G^\alpha \in C^\alpha. \end{aligned}$$

It is easy to see that the two codes $(C_0^\alpha,\, C_1^\alpha)$ give a direct sum decomposition of the window code $C^\alpha$.

Of course, there are numerous possibilities to decompose a given window code. However, we shall focus on the direct sum decomposition defined above. The decomposition is essential for the convolutional coding scheme discussed later. We also introduce the following encoding maps corresponding to $C_0^\alpha$ and $C_1^\alpha$:

$$\begin{aligned} \phi_0 \colon \quad & F_2^\gamma \to F_2^r, \\ \phi_0(\boldsymbol{x}) \quad & \mapsto (\mathbf{0}^{pm} | \boldsymbol{x} M_0) G^\alpha, \\ \phi_1 \colon \quad & F_2^{pm} \times F_2^\beta \to F_2^r, \\ \phi_1(\boldsymbol{v},\, \boldsymbol{w}) \quad & \mapsto (\boldsymbol{v} | \boldsymbol{w} M_1) G^\alpha. \end{aligned}$$

## *B. Details on DC-Free Convolutional Coding*

Assume that a convolutional code $C$ together with the parameters $\alpha,\, \beta,\, \gamma$, and a decomposition matrix $M$ are given. We call the code $C$ the *base convolutional code*. The following is the detail of the DC-free convolutional coding such as encoding, decoding, and its RDS bound.

*1) Encoding:* We first divide the message sequence $(u_0 u_1 u_2 \cdots)$ into blocks of length $\beta$. The $i$th $(i = 0,\, 1,\, 2,\, \ldots)$ message block is denoted by

$$\boldsymbol{u}_i = \left(u_{i\beta},\, u_{i\beta+1},\, u_{i\beta+2S},\, \ldots,\, u_{(i+1)\beta-1}\right).$$

The message sequences are encoded to the intermediate sequences by the RDS control encoder (cf. Fig. 1). We divide the intermediate sequence $(x_0 x_1 x_2 \cdots)$ into the intermediate blocks of length $\ell$. The $i$th intermediate block is defined by

$$\boldsymbol{x}_i = \left(x_{(\gamma+\beta)i},\, x_{(\gamma+\beta)i+1},\, \ldots x_{(\gamma+\beta)i+\ell-1}\right), \tag{15}$$

$$= (\boldsymbol{o}_i | \boldsymbol{n}_i) \tag{16}$$

where $\boldsymbol{o}_i$ is the first $pm$-tuple of $\boldsymbol{x}_i$ such that

$$\boldsymbol{o}_i = \left(x_{(\gamma+\beta)i},\, x_{(\gamma+\beta)i+1},\, \ldots x_{(\gamma+\beta)i+pm-1}\right) \tag{17}$$

and $\boldsymbol{n}_i$ is the last $\gamma + \beta$-tuple of $\boldsymbol{x}_i$ such that

$$\boldsymbol{n}_i = \left(x_{(\gamma+\beta)i+pm},\, x_{(\gamma+\beta)i+pm+1},\, \ldots x_{(\gamma+\beta)i+\ell-1}\right). \tag{18}$$

We obtain a coded sequence $(y_0 y_1, \ldots)$ by encoding the intermediate sequence with the convolutional encoder. The coded sequence is divided into the coded blocks of length $r$. The $i$th $(i = 0,\, 1,\, 2,\, \ldots)$ coded block has the form

$$\boldsymbol{y}_i = \left(y_{ri+qm},\, y_{ri+1+qm},\, \ldots y_{r(i+1)-1+qm}\right).$$

Note that $\boldsymbol{y}_0 = (y_{qm},\, \ldots,\, y_{r-1+qm})$. From (11), we have the relation between $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ such that $\boldsymbol{y}_i = \boldsymbol{x}_i G^\alpha$ $(i = 0,\, 1,\, \ldots)$. Figs. 3 and 4 might be helpful to understand the encoding procedure. Fig. 3 shows the relation between the message, intermediate and coded sequences and Fig. 4 illustrates the relation of the generator matrix of a convolutional code and a window matrix.

Notice that the intermediate blocks $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ are overlapping. The overlapping part corresponds to $\boldsymbol{o}_i$. By applying the

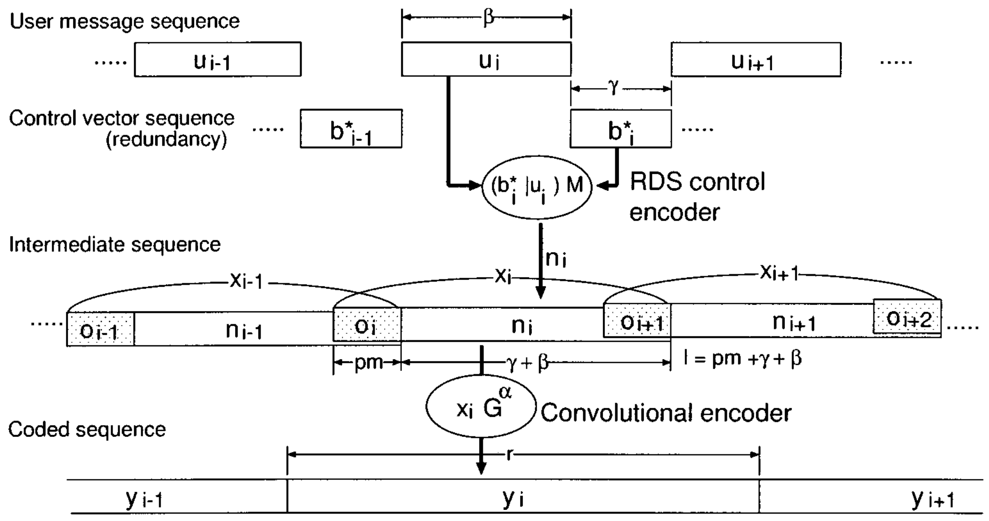
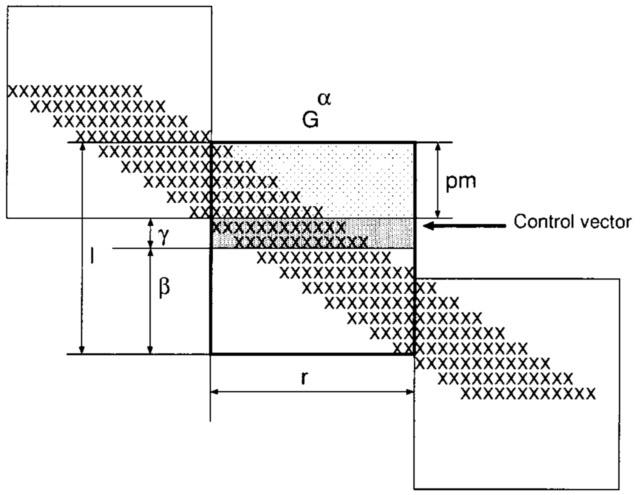


Fig. 3. Relation between message, intermediate and coded sequences.



The symbols "xxxx" mean binary sequences contained in the generator matrix of a convolutional code. For simplicity, a simple decomposition (described later) is assumed.

Fig. 4. Generator matrix and window matrix of a convolutional code.

additive encoder to a window code, we have to take the overlapping into account. Within the intermediate block $\boldsymbol{x}_i$, only the vector $\boldsymbol{n}_i$ can be assigned freely without any influence of the previous block. The overlapping part $\boldsymbol{o}_i$ is determined by the previous intermediate block $\boldsymbol{x}_{i-1}$. This is the reason why we assume a specific decomposition $(C_0^\alpha, C_1^\alpha)$ in (13) and (14).

As shown in Fig. 3, the RDS control encoder adds redundancy (a control vector) to the message sequence and thus the coding rate defined between the message and intermediate sequence becomes $\beta/(\gamma+\beta)$. The convolutional encoder of rate $p/q$ appends redundancy to the intermediate sequence. Consequently, the overall rate becomes

$$R \triangleq (p\beta)/(q(\gamma+\beta)).$$

The rate loss due to the RDS control encoder can be considered as the price for obtaining an RDS constraint.

The following completely describes how the RDS control encoder works.

*[RDS Control Encoder]:*

Step 1) (*Initialize*) Set RDS $:=$ RDS$_{\text{ini}}$, $i := 0$, and initial value of $\boldsymbol{o}_0 \in F_2^{pm}$ (the details about the initial values RDS$_{\text{ini}}$ and $\boldsymbol{o}_0$ will be discussed later).

Step 2) (*Control vector generation*) Generate the shell of $\phi_1(\boldsymbol{o}_i, \boldsymbol{u}_i)$

$$V_i := \{\phi_0(\boldsymbol{b}_i) \oplus \phi_1(\boldsymbol{o}_i, \boldsymbol{u}_i) : \boldsymbol{b}_i \in F_2^\gamma\}$$

and choose the best codeword $\boldsymbol{c}_i^* \in V_i$ according to rule-B. Let the vector $\boldsymbol{b}_i^*$ be the control vector satisfying $\boldsymbol{c}_i^* = \phi_0(\boldsymbol{b}_i^*) \oplus \phi_1(\boldsymbol{o}_i, \boldsymbol{u}_i)$.

Step 3) (*RDS update*) Set $RDS := RDS + S(\boldsymbol{c}_i^*)$.

Step 4) (*Output intermediate sequence*) Set $\boldsymbol{n}_i := (\boldsymbol{b}_i^*|\boldsymbol{u}_i)M$ and output $\boldsymbol{n}_i$ as a part of an intermediate sequence.

Step 5) (*Update memory*) Set $\boldsymbol{o}_{i+1} := \boldsymbol{n}_i^{(tail)}$, where $\boldsymbol{n}_i^{(tail)}$ is the last $pm$-tuple of $\boldsymbol{n}_i$.

Step 6) (*Counter increment*) Set $i := i + 1$ and return to Step 2).

The code $C_0^\alpha$ has $2^\gamma$-codewords. Thus, the RDS control encoder first generates $2^\gamma$-candidates of $\boldsymbol{c}_i^*$ and then selects the best one among the candidates according to Rule-B. This operation in Step 2) can be regarded as an additive encoder based on the decomposition $(C_0^\alpha, C_1^\alpha)$. When $\gamma$ is small enough such as $\gamma \le 4$, the computational task of the additive encoder seems to be very small and we could implement a high-speed RDS control encoder which is able to catch up with the encoding speed of the convolutional encoder.

The first $qm$-tuple of coded sequence

$$\boldsymbol{y}_{\text{ini}} \triangleq (y_0, y_1, \ldots, y_{qm-1})$$

exactly coincides with the first $qm$-tuple of $\boldsymbol{o}_0 G$. Thus $\boldsymbol{y}_{\text{ini}}$ depends only on $\boldsymbol{o}_0$. The initial value of the RDS, $\text{RDS}_{\text{ini}}$ appeared in Step 1), is given by

$$\text{RDS}_{\text{ini}} \triangleq \sum_{j=0}^{qm-1} f(y_j).$$

In order to obtain a tighter bound, we have to choose $\boldsymbol{o}_0$ that gives the smallest value of $|\text{RDS}_{\text{ini}}|$.

*2) Decoding:* We discuss the decoding issue for the proposed scheme. The received sequence is first decoded by the Viterbi decoder for the base convolutional code $C$. Let the set of all the allowable sequences generated by the proposed scheme be $C_{\text{RDS}}$. The minimum free Hamming distance defined on $C_{\text{RDS}}$ is denoted by $d'_{\text{free}}$. From the cascaded structure of the proposed scheme, evidently, $C_{\text{RDS}}$ is contained in $C$ and the inequality $d'_{\text{free}} \ge d_{\text{free}}$ holds. The symbol $d_{\text{free}}$ denotes the minimum free Hamming distance of $C$.

As a consequence of this property, we can use the Viterbi decoder for the base convolutional code to decode $C_{\text{RDS}}$. It can be considered as a kind of a super code decoding.

The decoding of the intermediate sequence is straightforward from the definition of the RDS control encoder. Let $\hat{\boldsymbol{b}}_i^*$, $\hat{\boldsymbol{n}}_i$, and $\hat{\boldsymbol{u}}_i$ be the estimated blocks corresponding to $\boldsymbol{b}_i^*$, $\boldsymbol{n}_i$, and $\boldsymbol{u}_i$, respectively. The details on the RDS control decoder are as follows.

*[RDS Control Decoder]:*

Step 1) (*Initialize*) $i := 0$

Step 2) (*Inverse matrix*) Left-multiplying the inverse matrix of $M$ by $\hat{\boldsymbol{n}}_i$, we have

$$\hat{\boldsymbol{n}}_i M^{-1} := \left(\hat{\boldsymbol{b}}_i^* | \hat{\boldsymbol{u}}_i\right). \tag{19}$$

Output $\hat{\boldsymbol{u}}_i$ as the $i$th estimated message block.

Step 3) (*Counter increment*) Set $i := i + 1$ and return to Step 2). □

*3) RDS Bounds:* The next lemma is the basis to prove upper and lower bounds on RDS.

*Lemma 2:* The equality $\boldsymbol{y}_i = \boldsymbol{c}_i^*$ holds for $i = 0, 1, 2, \ldots$.

*Proof:* Since we have

$$\boldsymbol{y}_i = \boldsymbol{x}_i G^\alpha = (\boldsymbol{o}_i | \boldsymbol{n}_i)\, G^\alpha, \qquad i = 0, 1, 2, \ldots \tag{20}$$

it is sufficient to show that $\boldsymbol{c}_i^* = (\boldsymbol{o}_i|\boldsymbol{n}_i)G^\alpha$ for proving the claim of the lemma. From Step 4) of the encoding procedure and the definition of $M$, we obtain the following relation:

$$\boldsymbol{n}_i = \left(\boldsymbol{b}_i^* | \boldsymbol{u}_i\right) M = \left(\boldsymbol{b}_i^* | \boldsymbol{u}_i\right) \begin{pmatrix} M_0 \\ M_1 \end{pmatrix} = \boldsymbol{b}_i^* M_0 \oplus \boldsymbol{u}_i M_1. \tag{21}$$

By using the relation and the definitions of $\phi_0$, $\phi_1$, we immediately have

$$\begin{aligned}(\boldsymbol{o}_i|\boldsymbol{n}_i)\, G^\alpha &= \left(\boldsymbol{o}_i | \boldsymbol{b}_i^* M_0 \oplus \boldsymbol{u}_i M_1\right) G^\alpha \\ &= \left(0^{pm} | \boldsymbol{b}_i^* M_0\right) G^\alpha \oplus (\boldsymbol{o}_i | \boldsymbol{u}_i M_1)\, G^\alpha \\ &= \phi_0(\boldsymbol{b}_i^*) \oplus \phi_1(\boldsymbol{o}_i, \boldsymbol{u}_i) \\ &= \boldsymbol{c}_i^*. \end{aligned} \tag{22}$$

□

By applying the bound in Theorem 1 to the decomposition $(C_0^\alpha, C_1^\alpha)$, we can derive the upper and lower bounds on the RDS of the coded sequence.

*Theorem 2 (RDS Bound):* If

$$\mathcal{L}(C_0^\alpha, C_1^\alpha) \le \text{RDS}_{\text{ini}} \le \mathcal{U}(C_0^\alpha, C_1^\alpha)$$

then the RDS of coded sequence $z_t = \sum_{j=0}^{t} f(y_j)$ is bounded by $\mathcal{L}(C_0^\alpha, C_1^\alpha) \le z_t \le \mathcal{U}(C_0^\alpha, C_1^\alpha)$ for any time instant $t \ge 0$.

*Proof:* From Lemma 2, we have $\boldsymbol{y}_i = \boldsymbol{c}_i^*$ $(i = 0, 1, 2 \ldots)$. Each vector $\boldsymbol{c}_i^*$ is specified according to Rule-B. Therefore, we can use essentially the same argument of Theorem 1 in the proof. We thus need the condition

$$\mathcal{L}(C_0^\alpha, C_1^\alpha) \le \text{RDS}_{\text{ini}} \le \mathcal{U}(C_0^\alpha, C_1^\alpha)$$

to guarantee the claim of the theorem. □

*Example 5:* We present an example of an encoding procedure. Assume that we have the convolutional code from Example 4, which is the four-state rate–1/2 convolutional code with $d_{\text{free}} = 5$. For $\alpha = 0$, we have

$$G^0 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

We also assume that $\gamma = 1$, $\beta = 2$, and

$$M = \left(\frac{M_0}{M_1}\right) = \left(\begin{array}{ccc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right).$$

$\boldsymbol{o}_0 = (01)$ is chosen as an initial sequence, which gives $\text{RDS}_{\text{ini}} = 0$.

A user message sequence

$$(\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3, \ldots) = (10, 10, 01, 00, \ldots)$$

is assumed to be encoded. When the block index $i = 0$, the RDS control encoder selects $\boldsymbol{b}_0^* = 1$ as a control vector. We thus have $\boldsymbol{n}_0 = (\boldsymbol{b}_0^* | \boldsymbol{u}_0)M = (110)$, and $\boldsymbol{x}_0 = (\boldsymbol{o}_0 | \boldsymbol{n}_0) = (01110)$. In the same way, we have the following sequences: $\boldsymbol{b}_1^* = 0, \boldsymbol{b}_2^* = 1$, $\boldsymbol{b}_3^* = 0, \ldots$, and

$$\boldsymbol{x}_1 = (10010) \quad \boldsymbol{x}_2 = (10101) \quad \boldsymbol{x}_3 = (01000), \ldots. \tag{23}$$

Note that the intermediate sequences (23) become $(10010101000\cdots)$. The coded sequence obtained from the intermediate sequence is given by

$$(\boldsymbol{y}_{\text{ini}}, \boldsymbol{y}_0, \boldsymbol{y}_1, \boldsymbol{y}_2, \ldots) = (0011, 011001, 111110, 001000, \ldots). \tag{24}$$

In this case, the overall coding rate becomes $1/2 \times 2/(1+2) = 2/6$ and the upper and lower bounds on the RDS are given by $\mathcal{L}(C_0^\alpha, C_1^\alpha) = -8$ and $\mathcal{U}(C_0^\alpha, C_1^\alpha) = +6$. □

### C. Computer Search for Finding Good Decompositions

For a given window code $C^\alpha$, we need a good decomposition of $C^\alpha$ for achieving a tight RDS constraint. We present here a computer search method and good decompositions obtained by the exhaustive computer searches.

*1) Computer Search:* Let $I$ be the $(\beta+\gamma) \times (\beta+\gamma)$-identity matrix and $\boldsymbol{i}_j$ $(j = 0, 1, \ldots, \beta+\gamma-1)$ be the $j$th-row vector of $I$. From the definition of $C_0^\alpha$ and $C_1^\alpha$ ((13) and (14)), we can see that a direct sum decomposition can be completely described by a decomposition matrix $M$. In other words, we have to look for a decomposition matrix $M$ which gives a small absolute value of the RDS for a given window code. We restrict our attention to the case where $M$ is obtained from a row permutation of $I$. The restriction helps to reduce the number of possible candidates for $M$ and it makes the computer searches rather easy. We call the decomposition the *simple decomposition*. The restriction also leads to the simplest RDS control encoder/decoder because, if $M$ has the above property, the computation of $\boldsymbol{x}M_0$, $\boldsymbol{x}M_1$, or $\boldsymbol{x}M$ becomes much simpler. Furthermore, with the simple decomposition, $M^{-1}$ also becomes the row permuted version of the identity matrix. Thus, no error propagation occurs in the decoding process of the intermediate sequence in (19) when an estimated intermediate sequence contains bit errors. In this case, we can expect that its bit-error probability is at least as good as the bit-error probability obtained by the combination of the base convolutional code and the corresponding Viterbi decoder.

In order to describe a permuted matrix, we shall introduce some notation. Assume that a set of size $\gamma$

$$\Theta^0 \triangleq \{\theta_0^0, \theta_1^0, \ldots, \theta_{\gamma-1}^0\} \subset \{0, 1, 2, \ldots, \beta+\gamma-1\} \tag{25}$$

is given. The set is called the *controller position set*. Then the *message position set* of size $\beta$ is defined by

$$\Theta^1 \triangleq \{\theta_0^1, \theta_1^1, \ldots, \theta_{\beta-1}^1\} = \{0, 1, 2, \ldots, \beta+\gamma-1\} \backslash \Theta^0. \tag{26}$$

By using the sets defined above, we let $M_0$ and $M_1$ be

$$M_0 = \begin{pmatrix} \boldsymbol{i}_{\theta_0^0} \\ \boldsymbol{i}_{\theta_1^0} \\ \vdots \\ \boldsymbol{i}_{\theta_{\gamma-1}^0} \end{pmatrix} \qquad M_1 = \begin{pmatrix} \boldsymbol{i}_{\theta_0^1} \\ \boldsymbol{i}_{\theta_1^1} \\ \vdots \\ \boldsymbol{i}_{\theta_{\beta-1}^1} \end{pmatrix}. \tag{27}$$

TABLE IV
RDS CONTROL ENCODERS FOR RATE-1/2 FOUR-STATE CONVOLUTIONAL CODE
$p/q = 1/2, d_{free} = 5, m = 2, G_0 = 5, G_1 = 7$

| $R$ | $\mathcal{L}$ | $\mathcal{U}$ | $\alpha$ | $\gamma$ | BCPS |
|---|---|---|---|---|---|
| 2/6 | −9 | +6 | 0 | 1 | {2} |
| 1/6 | −5 | +3 | 0 | 2 | {2,3} |
| 2/8 | −4 | +4 | 1 | 2 | {2,4} |
| 1/8 | −3 | +2 | 1 | 3 | {2,3,4} |
| 3/10 | −7 | +7 | 2 | 2 | {2,4} |
| 2/10 | −4 | +3 | 2 | 3 | {3,4,5} |
| 4/12 | −9 | +11 | 3 | 2 | {2,5} |
| 3/12 | −5 | +5 | 3 | 3 | {2,4,6} |
| 2/12 | −3 | +3 | 3 | 4 | {3,4,5,6} |

TABLE V
RDS CONTROL ENCODERS FOR RATE-1/2 16-STATE CONVOLUTIONAL CODE
$p/q = 1/2, d_{free} = 7, m = 4, G_0 = 46, G_1 = 72$

| $R$ | $\mathcal{L}$ | $\mathcal{U}$ | $\alpha$ | $\gamma$ | BCPS |
|---|---|---|---|---|---|
| 3/10 | −8 | +9 | 0 | 2 | {4,7} |
| 2/10 | −5 | +5 | 0 | 3 | {4,7,8} |
| 1/10 | −4 | +4 | 0 | 4 | {4,5,7,8} |
| 4/12 | −12 | +12 | 1 | 2 | {4,6} |
| 3/12 | −5 | +7 | 1 | 3 | {4,6,8} |
| 2/12 | −5 | +5 | 1 | 4 | {4,5,6,7} |
| 5/14 | −14 | +12 | 2 | 2 | {4,6} |
| 4/14 | −8 | +7 | 2 | 3 | {4,7,9} |
| 3/14 | −6 | +6 | 2 | 4 | {4,5,7,8} |

TABLE VI
RDS CONTROL ENCODERS FOR RATE-1/2 64-STATE CONVOLUTIONAL CODE
$p/q = 1/2, d_{free} = 10, m = 6, G_0 = 554, G_1 = 744$

| $R$ | $\mathcal{L}$ | $\mathcal{U}$ | $\alpha$ | $\gamma$ | BCPS |
|---|---|---|---|---|---|
| 5/14 | −13 | +13 | 0 | 2 | {6,10} |
| 4/14 | −8 | +8 | 0 | 3 | {6,7,9} |
| 3/14 | −7 | +7 | 0 | 4 | {6,7,8,9} |
| 6/16 | −18 | +18 | 1 | 2 | {6,10} |
| 5/16 | −9 | +9 | 1 | 3 | {7,9,12} |
| 4/16 | −7 | +7 | 1 | 4 | {6,7,8,9} |
| 6/18 | −11 | +11 | 2 | 3 | {6,9,10} |
| 5/18 | −8 | +8 | 2 | 4 | {6,7,8,9} |

It is easy to see that the above definitions of $M_0$ and $M_1$ correspond to $M$ which is obtained from the identity matrix by a row permutation.

The objective of the computer searches is to find a controller position set which gives the smallest value of an upper bound on the digital sum variation defined by

$$\Delta \triangleq \mathcal{U}(C_0^\alpha, C_1^\alpha) - \mathcal{L}(C_0^\alpha, C_1^\alpha). \tag{28}$$

The computer search algorithm is the following. For a given base convolutional code $C$, and parameters $\alpha$, $\beta$, $\gamma$, we first derive the window matrix $G^\alpha$. We then generate all the possible controller position sets of size $\gamma$ sequentially. For each controller position set, $\Delta$ is computed by making use of Theorem 2. Finally, the controller position set which gives the minimum $\Delta$ is chosen as the best controller position set.

The computer search results are summarized in Tables IV–VI. The best binary convolutional codes listed in the book by Lin and Costello [16] have been used as the base convolutional codes. In these tables, "BCPS" means the "best controller position set" and the symbol $G_i$ denotes the generator polynomial of the base convolutional code in octal notation from [16].

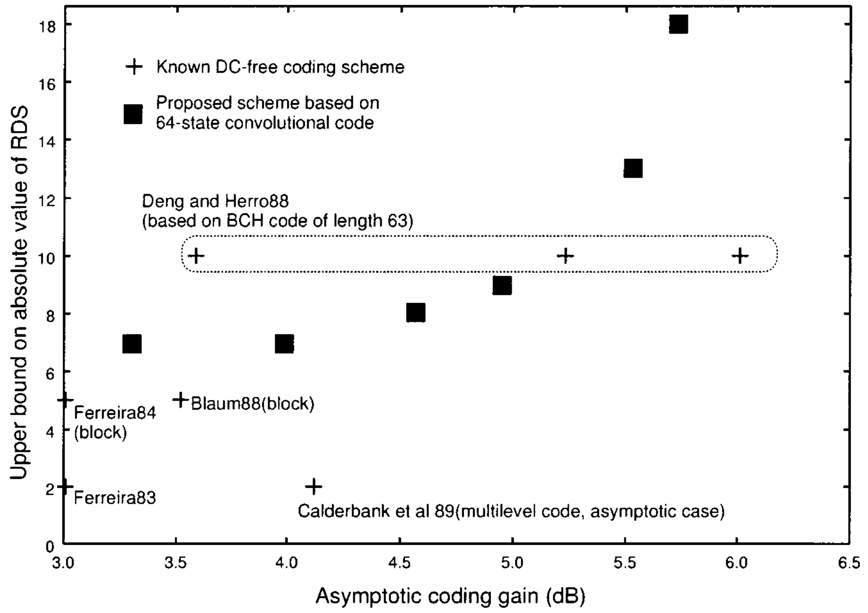


Fig. 5. Tradeoffs between asymptotic coding gain and RDS bound.

Of course, there might exist better convolutional encoders for our purpose. However, the exhaustive search considering combination of the convolutional encoder and its simple decomposition requires an enormous number of computations and is considered to be infeasible. We therefore fixed a base convolutional code in our computer search.

*2) Comparison:* In Fig. 5, the tradeoffs between the asymptotic coding gain (ACG) and the upper bound on the absolute value of the RDS for the proposed scheme based on the 64-state base convolutional code are presented. Parameters of several known DC-free coding schemes are also plotted in Fig. 5 for comparison. The ACG defined by ACG $\triangleq 10\log_{10}(Rd_{\text{free}})$ means the coding gain compared with uncoded signals over an additive white Gaussian channel when the signal-to-noise ratio (SNR) goes to infinity.

Most DC-free block coding schemes have better tradeoffs in the low coding gain area (less than 4.5 dB) compared with the proposed scheme but the difference is not large. We can see that considerably high coding gains (up to 5.7 dB) can be obtained by the proposed scheme. In the high coding gain area (more than 4.5 dB), few DC-free coding schemes with reasonable decoding complexity seem to be known. It is fair to mention that Deng and Herro's coding schemes also achieve high coding gains. However, their code is based on the BCH code of length 63 and thus the code requires a much more complex Viterbi decoder than the one proposed in this paper for attaining the maximum-likelihood decoding performance. It can be said that the proposed scheme gives us a wide range of varieties of reasonable tradeoffs between the coding gain, the RDS constraint, and decoding complexity.

## D. Simulation Results

In order to verify the performance of the proposed scheme, we have performed encoding simulations. In an encoding simulation, randomly generated message sequences are encoded by the RDS control encoder and the convolutional encoder. The conditions and the parameters related to the simulations are as follows. The base convolutional code is the rate–$1/2$ 64-state convolutional code $(m = 6, G_0 = 554, G_1 = 744)$ with $d_{\text{free}} = 10$. Table VII presents the results. The symbols $L$ and $U$ denote the minimum and the maximum RDS observed in an encoding simulation. It is known that the sum variance $E[z_t^2]$ is closely related to the near-DC-suppression characteristic of a DC-free coding scheme [1], where $E[\cdot]$ means the expectation value. We here define the sample sum variance $S^2$ by

TABLE VII
SIMULATION RESULTS FOR DC-FREE CONVOLUTIONAL ENCODING: BASE CONVOLUTIONAL CODE IS RATE–$1/2$ 64-STATE CONVOLUTIONAL CODE WITH $d_{\text{free}} = 10$

| $R$ | $\mathcal{L}$ | $\mathcal{U}$ | $L$ | $U$ | $S^2$ | $ACG$(dB) |
|---|---|---|---|---|---|---|
| $5/14$ | $-13$ | $+13$ | $-11$ | $+13$ | 7.83 | 5.53 |
| $4/14$ | $-8$ | $+8$ | $-6$ | $+8$ | 3.56 | 4.56 |
| $3/14$ | $-7$ | $+7$ | $-7$ | $+5$ | 2.78 | 3.31 |
| $6/16$ | $-18$ | $+18$ | $-12$ | $+13$ | 8.33 | 5.74 |
| $5/16$ | $-9$ | $+9$ | $-7$ | $+9$ | 4.51 | 4.95 |
| $4/16$ | $-7$ | $+7$ | $-5$ | $+7$ | 2.92 | 3.98 |

$$S^2 \triangleq \frac{1}{N}\sum_{t=0}^{N-1} z_t^2 \tag{29}$$

where $N$ is the number of samples. From Table VII, we can see that the values $L$ and $U$ are certainly within the range $[\mathcal{L}(C_0^\alpha, C_1^\alpha), \mathcal{U}(C_0^\alpha, C_1^\alpha)]$. It can also be recognized that fairly smaller values of a sample sum variance $S^2$ are attained by the proposed scheme compared with the upper and lower bounds on the RDS.

For evaluating near DC-suppression characteristics, we have also computed the power spectrum values for the coded sequence $(y_0, y_1, \ldots)$

$$A(\omega_i) \triangleq \frac{1}{N}\left|\sum_{t=0}^{N-1} f(y_t)\exp(-jt\omega_i)\right|$$

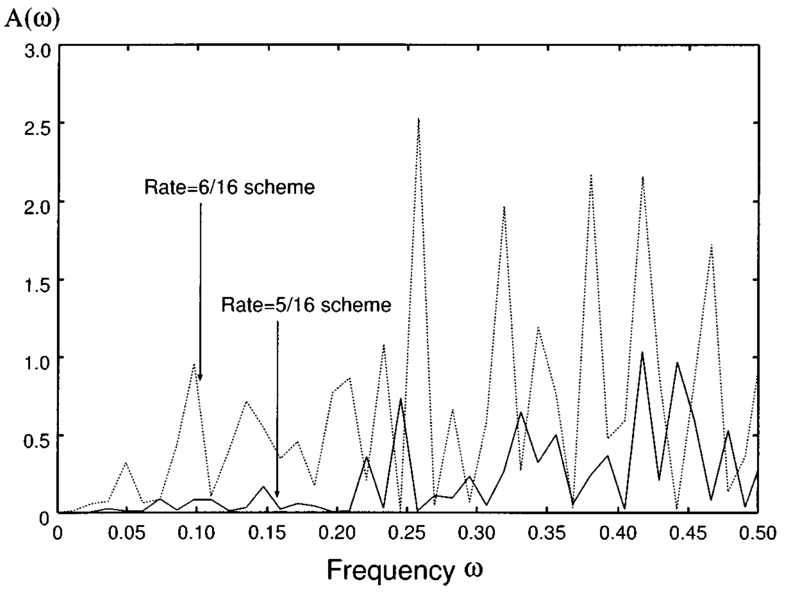


Fig. 6. Power spectrum characteristic of the proposed scheme at near DC.

where $\omega_i = (\pi/256)i$. The results for the $R = 6/16$ and the $R = 5/16$ schemes (in Table VII) are presented in Fig. 6. We observe that both schemes achieve DC-free and near-DC suppression. Especially, the $R = 5/16$ scheme shows rather better near-DC-suppression characteristic than the $R = 6/16$ scheme. Note that the $R = 6/16$ and the $R = 5/16$ schemes give sum variances $8.33$ and $4.51$, respectively.

## IV. Conclusion

In this paper, a new DC-free convolutional coding scheme has been presented. The scheme is suitable for a power-limited noisy channel. Availability of soft-decision decoding is one of the major advantages of the proposed scheme. By using the RDS bound derived in the paper, we can guarantee the RDS bound for the proposed scheme explicitly.

The proposed scheme can be divided into two parts; the RDS control encoder/decoder and the convolutional encoder/decoder. The RDS control encoder generates only several codewords of a window code for selecting a control vector. The decoder requires much simpler tasks than the encoder. Therefore, the RDS control encoder/decoder seems to be simple enough to be implemented.

As shown in the search results of Tables IV–VI, some good RDS control encoders have been found successfully by computer searches. For example, a 64-state DC-free coding scheme with overall rate $6/16$ and minimum free distance $10$ satisfies a bounded RDS condition (from $-18$ to $+18$) and yields the asymptotic coding gain (over an AWGN channel) of 5.7 dB. Furthermore, the proposed scheme offers a system designer numerous choices of reasonable tradeoffs between the error correcting capability (coding gains), the RDS constraint, and the decoding complexity.

In addition, the proposed scheme looks promising as a component code in a concatenated code such as serial concatenated coding. This is because the code sequences generated by the proposed scheme are contained in the set of code sequences generated by a convolutional encoder. Further coding gains could be obtained from such a concatenated scheme even for a channel with a low SNR.

The proposed scheme has been designed according to the following simple design principle: first, we prove fundamental properties on an additive encoder and then extend the result to the window code obtained from a base convolutional code. We expect that the principle can be applied to a convolutional coding with another constraint such as a run length constraint.

## Acknowledgment

The authors wish to thank the anonymous reviewers for helpful comments.

## References

[1] K. A. S. Immink, *Coding Techniques for Digital Recorders*. Englewood Cliffs, NJ: Prentice-Hall, 1991.
[2] H. C. Ferreira, "On DC free magnetic recording codes generated by finite state machines," *IEEE Trans. Magn.*, vol. MAG-19, pp. 2691–2693, 1983.
[3] ——, "Lower bounds on the minimum Hamming distance achievable with runlength constrained or DC free block codes and the synthesis of a (16, 8) $D_{\min} = 4$ DC free block code," *IEEE Trans. Magn.*, vol. MAG-20, pp. 881–883, 1984.
[4] M. Blaum, "A (16,9,6,5,4) error-correcting DC free block code," *IEEE Trans. Inform. Theory*, vol. 34, pp. 138–141, Jan. 1988.
[5] R. H. Deng and M. A. Herro, "DC-free coset codes," *IEEE Trans. Inform. Theory*, vol. 34, pp. 786–792, July 1988.
[6] A. R. Calderbank, M. A. Herro, and V. Telang, "A multilevel approach to the design of DC-free line codes," *IEEE Trans. Inform. Theory*, vol. 35, pp. 579–583, May 1989.
[7] A. R. Calderbank, T. A. Lee, and J. E. Mazo, "Baseband trellis codes with a spectral null at zero," *IEEE Trans. Inform. Theory*, vol. 34, pp. 425–434, May 1988.
[8] G. D. Forney, Jr. and A. R. Calderbank, "Coset codes for partial response channel; or, coset codes with spectral nulls," *IEEE Trans. Inform. Theory*, vol. 35, pp. 925–943, Sept. 1989.
[9] D. Kim and M. V. Eyuboglu, "Convolutional spectral shaping," *IEEE Commun. Lett.*, vol. 3, pp. 9–11, Jan. 1999.
[10] R. H. Deng, Y. X. Li, and M. A. Herro, "DC-free error correcting convolutional codes," *Electron. Lett.*, vol. 29, pp. 1910–1911, 1993.

[11] M. Nasiri-Kenari and C. K. Rushforth, “A class of DC-free subcodes of convolutional codes,” *IEEE Trans. Commun.*, vol. 44, pp. 1389–1391, 1996.
[12] M. C. Chiu, “DC-free error correcting codes based on convolutional codes,” in *Proc. 2000 IEEE Int. Symp. Information Theory*, Sorrento, Italy, 2000, pp. 25–30.
[13] J. M. Borden and A. J. H. Vinck, “On coding for ‘stuck-at’ defects,” *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 729–735, Sept. 1987.
[14] I. J. Fair, W. D. Grover, W. A. Krzmien, and R. I. MacDonald, “Guided scrambling: A new line coding technique for high bit rate fiber optic transmission systems,” *IEEE Trans. Commun.*, vol. 39, pp. 289–297, Feb. 1991.
[15] K. A. S. Immink, “Performance assessment of DC-free multimode codes,” *IEEE Trans. Commun.*, vol. 45, pp. 293–299, Mar. 1997.
[16] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.