SPECTRUM SHAPING WITH BINARY DC^2-CONSTRAINED CHANNEL CODES

Abstract

A method is presented for designing binary channel codes in such a way that both the power spectral density function and its second-derivative vanish at zero frequency. Recursion relations are derived to determine the number of codewords that can be used in this coding scheme. A simple algorithm for encoding and decoding codewords is developed. The performance of the new codes is compared with that of classical channel codes designed with a constraint on the unbalance of the number of transmitted positive and negative pulses.

1. Introduction

Many channel codes have been designed with the aim of suppressing the power of the encoded stream near the zero frequency. Most of these channel codes are designed on a digital sum constraint, where the (running) digital sum is defined for a binary sequence as the accumulated sum of ones and zeros (a zero is counted as -1) from the start of the transmission. The frequency region with suppressed components is characterized by the so-called cut-off frequency. Justesen[1]) found that the cut-off frequency of these sequences is proportional to the redundancy of the code. An example of a practical embodiment of a code rejecting the low-frequency components, a DC-free or DC-balanced code, is the so-called zero-disparity code using codewords with an equal number of positive and negative pulses[2]). A generalization of this concept leads to low-disparity codes using two alternative source representations of opposite disparity polarity[3]). Immink[4]) studied the cut-off frequency versus the redundancy of these simple binary low-disparity codes. He concluded that the performance of the codes, in particular for small codeword lengths, is comparable with that of maxentropic sum constrained sequences. The power spectral density function of digital sum constrained codes is characterized by a parabolic shape in the low-frequency range from DC to the cut-off frequency. In some applications it is desirable to achieve for a given redundancy a larger rejection of the low-frequency components than is possible with DC-balanced codes.

In this paper a new class of DC-free codes is presented having as additional property that the zero second-derivative of the code spectrum also vanishes at zero frequency. (Note that the odd derivatives of the spectrum at zero frequency are zero because the spectrum is an even function of the frequency.) This results in a substantial decrease of the power at low frequencies for a fixed code redundancy as compared with the classical designs based on a digital sum criterion. Section 2 introduces a time domain constraint to be imposed on the channel sequence so that the resulting spectrum of the sequence has the DC^2-balanced property, i.e. has both zero power and zero second-derivative at zero frequency. Section 3 presents an enumeration method for finding the number of codewords to be used in a DC^2-balanced code. In order for the new codes to have a practical rate relatively long codewords have to be used which makes a direct method of encoding and decoding using look-up tables of the source words and their channel representations prohibitively complex. Section 4 deals with the enumerative encoding and decoding of DC^2-balanced codewords. The algorithm is not more complex than looking-up and adding. The look-up tables needed for the enumerative coding grow polynomially in complexity with increasing codeword length. Section 5 gives examples of codes using codewords that can be concatenated without a merging rule, so-called state-independent encoding. The power spectral density functions of the newly developed codes will be compared with those of classical DC-balanced codes. Examples of codes with state-dependent encoding that operate with two modes of the source representation are given in sec. 6.

2. DC^2-balanced codes

Let $\boldsymbol{x}$ denote a stationary channel sequence, having mean zero, with variables $\ldots, x_{-1}, x_0, x_1, \ldots; x_i \in \{-1, 1\}$. The power spectral density function of the process is given by [5])

$$H(\omega) = R(0) + 2 \sum_{i=1}^{\infty} R(i) \cos(i\,\omega), \tag{1}$$

where $R(i) = E(x_j x_{j+i})$, $i = 0, \pm 1, \pm 2, \ldots$ is the auto-correlation function of the sequence.

The spectrum $H(\omega)$ is an even function of ω, so that in the neighbourhood of $\omega = 0$ it can be approximated with the following Taylor expansion:

$$H(\omega) = H(0) + \frac{H''(0)}{2}\omega^2 + \frac{H^{(iv)}(0)}{24}\omega^4 \ldots . \tag{2}$$

From a straightforward Fourier analysis it can be shown, if the running digital sum (RDS) z_i defined by

$$z_i = \sum_{j=-\infty}^{i} x_j,$$

is bounded, i.e. has finite unbalance, the resulting power spectral density function vanishes at $\omega = 0$, i.e. $H(0) = 0$ [6]).

We now define the running digital sum sum(RDSS) y_i by

$$y_i = \sum_{j=-\infty}^{i} z_j.$$

If both the RDS and RDSS assume a limited number of values it can be proved in the same way that then $H(0) = H''(0) = 0$. In the following a method is presented to design codes with bounded RDS and RDSS, here called 'DC²-balanced' codes.

The codes are based on codewords with fixed length. The codewords are chosen in such a way that if concatenated using a certain encoder rule the channel sequence will assume a finite number of y and z values. The first problem to solve is to enumerate the number of codewords with length n satisfying a y and z constraint.

3. Enumeration of (z,y) sequences

In the preceding section the word unbalance referred to a property of a sequence of arbitrary length. Now we shall use this concept for codewords with a predefined length. First simple recursion relations are derived to find the number of codewords with length n having a constraint on the z and y disparity, so-called (z,y) sequences. Later we show how these codewords can be concatenated in such a way that the code stream will assume a limited number of RDS and RDSS values.

In other words, find the number of sequences $\boldsymbol{x} = (x_1, \ldots, x_n)$, $x_i \in \{-1, 1\}$, satisfying the following conditions

$$\sum_{i=1}^{n} x_i = d_z \quad \text{and} \quad \sum_{j=1}^{n} \sum_{i=1}^{j} x_i = d_y, \tag{3}$$

where d_z and d_y are the z and y disparities of $\boldsymbol{x}$.

We follow the approach of [7]). By changing the order of summation we find

$$d_y = \sum_{i=1}^{n} (n+1-i)\, x_i = -\sum_{i=1}^{n} i\, x_i + (n+1)\, d_z. \tag{4}$$

Because $x_i \in \{-1, 1\}$ we rewrite eq. (4) and find

$$d_y = -\sum_{i=1}^{\frac{1}{2}(n+d_z)} p_i + \sum_{i=1}^{\frac{1}{2}(n-d_z)} n_i + (n+1)\,d_z, \tag{5}$$

where $p_i \in \{1, \ldots, n\}$ and $n_i \in \{1, \ldots, n\}$ correspond to the positions of the +1's and −1's, respectively.

Obviously

$$\sum_{i=1}^{\frac{1}{2}(n+d_z)} p_i + \sum_{i=1}^{\frac{1}{2}(n-d_z)} n_i = \sum_{i=1}^{n} i = \tfrac{1}{2}n(n+1),$$

so that with eq. (5)

$$\sum_{i=1}^{\frac{1}{2}(n+d_z)} p_i = \tfrac{1}{4}[n(n+1) + 2(n+1)\,d_z - 2d_y]. \tag{6}$$

Remark

In the particular case that both z and y disparities are zero it follows in a straightforward way from eqs (3) and (6) that the codeword length n has to be a multiple of four.

The number of codewords satisfying the d_y and d_z constraints can be found using the theory of partitions (see ref. 8). From the above it follows that this number equals the number of distinct subsets $\{p_1, \ldots, p_{\frac{1}{2}(n+d_z)}\}$ of size $\frac{1}{2}(n+d_z)$ of the set $\{1, 2, \ldots, n\}$ satisfying eq. (6). The number of codewords, denoted by $A_n(d_z, d_y)$, is given by the coefficient $C_n(i,j)$ of $u^i t^j$, of the polynomial $g_n(u,t)$ defined by

$$g_n(u,t) = (1+ut)(1+u^2t)\ldots(1+u^nt),$$

where

$$i = \tfrac{1}{4}[n(n+1) + 2(n+1)\,d_z - 2d_y], \quad j = \tfrac{1}{2}(n+d_z).$$

Both i and j are assumed to be integers, otherwise the number of codewords is zero.

We can derive a useful relation with the following observation. The sequence $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ satisfies the d_z and d_y constraint if the sequence $\boldsymbol{w}$ obtained by reversing the order of symbols in $\boldsymbol{x}$, i.e. $\boldsymbol{w} = (x_n, \ldots, x_1)$, satisfies the constraints

$$\sum_{i=1}^{n} w_i = d_z, \quad \sum_{j=1}^{n} \sum_{i=1}^{j} w_i = -d_y + (n+1)\,d_z.$$

The number of sequences $\boldsymbol{w}$ satisfying the two constraints equals by definition

$A_n(d_z, (n+1)d_z - d_y)$. As the number of sequences $\boldsymbol{w}$ equals the number of sequences $\boldsymbol{x}$ with the given constraints we simply obtain

$$A_n(d_z, d_y) = A_n(d_z, (n+1)\, d_z - d_y). \tag{7}$$

As $g_n(u,t) = g_{n-1}(u,t)(1 + u^n t)$ the following recursion relation of the coefficients can be written down.
For $m = 1, 2, \ldots, n$; $j = 0, \ldots, m$ and $i = 0, \ldots, \frac{1}{2}m(m+1)$

$$C_m(i,j) = C_{m-1}(i,j) + C_{m-1}(i-m, j-1), \tag{8}$$

with initial conditions

$$C_0(0,0) = 1, \quad \text{otherwise} \quad C_m(i,j) = 0.$$

4. Enumerative encoding and decoding (z,y) sequences

Codes with rates of practical interest are only possible if the codeword length is relatively large (see later). The hardware requirements for a direct encoding and decoding using look-up tables for storing the codewords and their source representations grow exponentially with the codeword length. For say $n > 24$ this direct method is beyond current technology. In this section a sequential algorithm is presented for encoding and decoding (z,y) sequences which is not more complex than adding and looking-up, where the number of entries of the look-up table grows polynomially with the codeword length.

To that end we establish a $1-1$ mapping from the set S of (z,y) sequences onto the set of integers $0, 1, \ldots, M-1$, where M is the cardinality of S.

Let S be the set of (z,y) sequences. The set S can be ordered lexicographically as follows:

If $\boldsymbol{w} = (w_1, \ldots, w_n)$ and $\boldsymbol{v} = (v_1, \ldots, v_n)$ are elements of the set S then $\boldsymbol{v}$ is called less than $\boldsymbol{w}$, in short $\boldsymbol{v} <_0 \boldsymbol{w}$, if there exists an $i, 1 \leqslant i \leqslant n$, such that $v_i < w_i$ and $w_j = v_j$, $1 \leqslant j < i$.

The position of $\boldsymbol{w}$ in the lexicographical ordering of S is defined to be the rank of $\boldsymbol{w}$, denoted by $r(\boldsymbol{w})$, i.e. $r(\boldsymbol{w})$ is the number of all $\boldsymbol{v}$ in S with $\boldsymbol{v} <_0 \boldsymbol{w}$.

The following algorithm for ranking (z,y) sequences can be used for decoding.

Decoding algorithm

The rank $r(\boldsymbol{x})$ of the binary (z,y) sequence $\boldsymbol{x}$ in S can be calculated as follows

$$r(\boldsymbol{x}) = \sum_{k=1}^{n} A_{n-k}(d_z - z_{k-1} + 1, \; -d_y + (n-k+1)\, d_z + y_{k-1}) \cdot x_k',$$

where

$$x_k' = \tfrac{1}{2}(x_k + 1),$$

$$z_{k-1} = \sum_{i=1}^{k-1} x_i$$

and

$$y_{k-1} = \sum_{i=1}^{k-1} z_i.$$

Proof

According to Cover [9]) the lexicographic index $r(\boldsymbol{x})$ of $\boldsymbol{x}$ is given by

$$r(\boldsymbol{x}) = \sum_{j=1}^{n} x_j' \cdot n_s(x_1, x_2, \ldots, x_{j-1}, -1),$$

where $n_s(x_1,x_2,\ldots,x_k)$ denotes the number of elements in S for which the first k coordinates are given by $(x_1,x_2,\ldots,x_k)$. The first $(j-1)$ coordinates of $\boldsymbol{x}$ determine the states z_j and y_j

$$z_{j-1} = \sum_{i=1}^{j-1} x_i$$

and

$$y_{j-1} = \sum_{i=1}^{j-1} z_i.$$

From eq. (3) we find that

$$d_z = \sum_{i=1}^{j-1} x_i + x_j + \sum_{i=j+1}^{n} x_i = $$
$$= z_{j-1} + x_j + \sum_{i=1}^{n-j} x_{j+i}.$$

In the same way

$$d_y = \sum_{i=1}^{n} z_i = \sum_{i=1}^{j-1} z_i + z_j + \sum_{i=j+1}^{n} z_i =$$
$$= \sum_{i=1}^{j-1} z_i + z_j + \sum_{i=j+1}^{n} \sum_{m=1}^{i} x_m$$
$$= \sum_{i=1}^{j-1} z_i + (n-j+1)\, z_j + \sum_{i=j+1}^{n} \sum_{m=j+1}^{i} x_m$$
$$= \sum_{i=1}^{j-1} z_i + (n-j+1)\, z_j + \sum_{k=1}^{n-j} \sum_{m=1}^{k} x_{j+m}.$$

From these observations it follows that the number of vectors $\boldsymbol{x}$ given by

$$\boldsymbol{x} = (u_1, \ldots, u_{j-1}, -1, x_{j+1}, \ldots, x_n),$$

(where the u_i's are given) satisfying

$$\sum_{i=1}^{n} x_i = d_z \quad \text{and} \quad \sum_{i=1}^{n} \sum_{m=1}^{i} x_m = d_y$$

equals the number of vectors $\boldsymbol{v}$ with length $n-j$, $\boldsymbol{v} = (v_1, \ldots, v_{n-j})$ satisfying

$$\sum_{i=1}^{n-j} v_i = d_z - z_{j-1} + 1$$

and

$$\sum_{k=1}^{n-j} \sum_{m=1}^{k} v_m = d_y - \sum_{i=1}^{j-1} z_i - (n-j+1)\, z_{j-1} + (n-j+1).$$

Whence

$$n_s(x_1, x_2, \ldots, x_{j-1}, -1) =$$
$$A_{n-j}(d_z - z_{j-1} + 1, d_y - y_{j-1} + (n-j+1)(1 - z_{j-1})).$$

With eq. (7) we find

$$A_{n-j}(d_z - z_{j-1} + 1, d_y - y_{j-1} + (n-j+1)(1 - z_{j-1})) =$$
$$= A_{n-j}(d_z - z_{j-1} + 1, -d_y + (n-j+1)\, d_z + y_{j-1}).$$

□

The inverse algorithm to encode (z,y) sequences, i.e. to find the sequences given their rank, is given in the following.

Encoding algorithm

Let an integer I, $I \in \{0, 1, \ldots, M-1\}$ and the set S be given. The following algorithm finds $\boldsymbol{x}$ such that $r(\boldsymbol{x}) = I$.

Initialize

$$x_0 = y_0 = z_0 = 0;$$

for

$$k = 1, \ldots, n, \quad \textit{if} \quad A_{n-k}(d_z - z_k + 1, (n-k+1)\, d_z - d_y + y_k) \leqslant I,$$

where $z_k = z_{k-1} + x_{k-1}$ and $y_k = y_{k-1} + z_{k-1}$

then set $x_k = 1$ and *set*

$$I = I - A_{n-k}(d_y - z_k + 1, (n-k+1)\, d_z - d_y + y_k)$$

else set $x_k = -1$.

The procedure, as presented here, is confined to the encoding and decoding of one set of (z,y) sequences. The procedure can readily be generalized to include sets with different values of the disparities (see ref. 9).

The enumerative coding enables the design of channel encoders and decoders of moderate complexity. For decoding we need storage capacity for the non-zero weighting coefficients, a full adder and an accumulator to store intermediate and final results. The encoder needs an additional logical comparator. The weighting coefficients look-up table can be used for encoding and decoding, which is attractive for application in recording, where the transmitter and receiver are usually housed in one machine. Note that we do not need a multiplier because x is binary valued. Lyon[10]) gave an example of a practical embodiment based on the enumerative coding of codewords with given z disparity using Pascal's triangle. In order to encode and decode (z,y) sequences his circuitry has to be extended with an extra counter to determine the y state.

5. Coding with zero-disparity codewords

The simplest way to find a code generating long sequences satisfying the z and y constraint is to use codewords of fixed length n that can be concatenated without a coding rule, i.e. to use a stateless encoder. In this simple case the codewords start and terminate with zero RDS and RDSS. Consequently the codewords have zero z and y disparities. In the sequel we use the short-hand notation zero-disparity if both the z and y disparities are zero. From eqs (3) and (4) we find in this particular case that the codeword moments u_k satisfy the following conditions:

$$u_k = \sum_{i=1}^{n} i^k x_i = 0, \quad k \in \{0, 1\}. \tag{9}$$

Using the recursion relations eq. (8) we calculated the number $M = A_n(0,0)$, $n = 4, 8, \ldots, 36$ of zero-disparity codewords (we have already remarked that $A_n(0,0) = 0$, if n is not a multiple of four). The results are listed in table I.

TABLE I

Number of codewords M with zero-disparity

n	M	R
4	2	0.250
8	8	0.375
12	58	0.488
16	526	0.565
20	5448	0.621
24	61108	0.662
28	723354	0.695
32	8908546	0.722
36	113093022	0.743

Table I also presents the code rate R, defined by

$$R = \frac{{}^2\log M}{n}. \tag{10}$$

Note in table I that even for large codeword length the code rate is still poor.

During transmission the symbols of the (z,y) sequences are serially transmitted. Assume that the codewords are equiprobable and independent, then the auto-correlation function of the concatenated channel stream is given by [11])

$$R(k) = \frac{1}{n}\sum_{i=1}^{n-k} E(x_i x_{i+k}); \quad 1 \leqslant k \leqslant n-1,$$

$$R(0) = 1,$$

$$R(k) = 0; \quad k \geqslant n,$$

where

$$E(x_i x_j) = \frac{1}{M}\sum_{k=0}^{M-1} x_i^{(k)} x_j^{(k)}; \quad 1 \leqslant i,j \leqslant n. \tag{11}$$

where $x^{(k)}$ is the k-th element of the set of codewords with zero y and z disparity.

Example 1

The simplest example is the code with $n = 4$ having two codewords so that the rate $R = \frac{1}{4}$. We can verify that the two codewords are '0110' and '1001' (a '0' is used to denote a symbol with value -1). The auto-correlation function of this channel code can be found by hand: $R(0) = 1$, $R(1) = -\frac{1}{4}$, $R(2) = -\frac{1}{2}$, $R(3) = \frac{1}{4}$ and for $i \geqslant 4$: $R(i) = 0$. The resulting power spectral density function $H(\omega)$ of this channel code is found with eq. (1)

$$H(\omega) = 1 + 2\{-\tfrac{1}{4}\cos\omega - \tfrac{1}{2}\cos 2\omega + \tfrac{1}{4}\cos 3\omega\} = 4\sin^2\tfrac{1}{2}\omega\,\sin^2\omega.$$

We can easily verify that indeed $H(0) = H''(0) = 0$.

Due to the exponential growth of the number of codewords the computational effort becomes prohibitive for large codeword lengths when eq. (11) is used. The preceding enumeration method (sec. 3) can be extended to find the auto-correlation function with a polynomially growing computational effort. From eq. (11) we derive for fixed i_0, i_1; $(x_i \in \{-1, 1\})$:

$$E(x_{i_0} x_{i_1}) = \frac{1}{M}\{N(x_{i_0} = x_{i_1}) - N(x_{i_0} \neq x_{i_1})\}; \quad i_0 \neq i_1,$$

where $N(\ )$ is the number of codewords satisfying ( ).

For symmetry reasons we find

$$E(x_{i_0} x_{i_1}) = \frac{2N(x_{i_0} = x_{i_1})}{M} - 1$$
$$= \frac{4N(x_{i_0} = x_{i_1} = 1)}{M} - 1. \tag{12}$$

Assume $x_{i_0} = x_{i_1} = 1$, then from eq. (9) we find

$$\sum_{i=1,\, i \neq i_0,\, i_1}^{n} x_i = -2$$

and

$$\sum_{i=1,\, i \neq i_0,\, i_1}^{n} i\, x_i = -(i_0 + i_1). \tag{13}$$

The number of sequences in S satisfying these constraints can again be found using the enumeration method of sec. 3.

The number of sequences $N(x_{i_0} = x_{i_1} = 1)$ in S satisfying eq. (13) is given by the coefficient of $u^i v^j$, $i = \frac{1}{4}n(n+1) - i_0 - i_1$, $j = \frac{1}{2}(n-4)$ of the polynomial $h_n(u, t)$

$$h_n(u, t) = \frac{g_n(u, t)}{(1 + u^{i_0} t)(1 + u^{i_1} t)}. \tag{14}$$

With a small modification of the recursion relations given by eq. (8) we can obtain $N(x_{i_0} = x_{i_1} = 1)$.

Using eqs (1), (11), (12) and relation (14) we calculated the power spectral density functions of zero-disparity codeword based codes. The results of the computations are plotted in fig. 1.

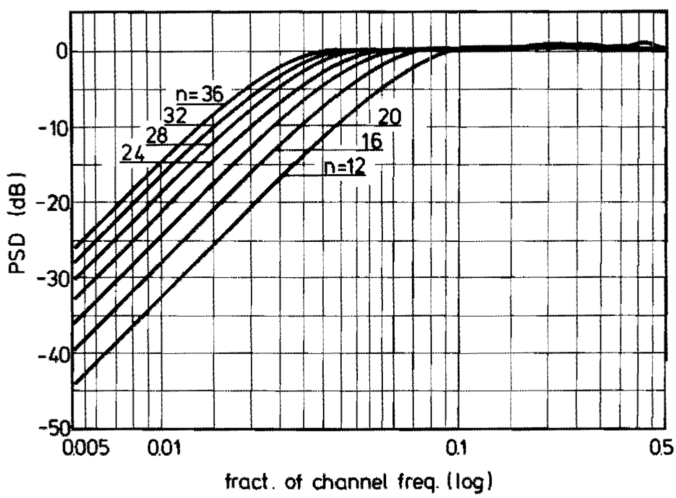

Fig. 1. Power spectral density functions of DC²-balanced codes based on zero-disparity codewords with the codeword length n as a parameter.

We define the (low-frequency) cut-off frequency ω_0 according to [12])

$$H(\omega_0) = \tfrac{1}{2}.$$

Using numerical methods we computed the cut-off frequency of the various codes. Fig. 2 shows the cut-off frequency versus the redundancy $1 - R$. As a comparison we plotted the cut-off frequency, according to the same definition, of DC-balanced zero-disparity codeword based codes versus the redundancy [12]). We observe in the figure that for a fixed rate the cut-off frequency of DC-balanced codes are approximately a factor of 2.5 larger than those of DC^2-balanced codes. Below the cut-off frequency the spectra of DC^2-balanced codes decreases faster with decreasing frequency than those of DC-balanced codes. We noticed when comparing the spectra of DC^2- and DC-balanced codes that for fixed redundancy a cross-over is found at approximately -20 dB. We conclude therefore that DC^2-balanced codes are favourable if rejection of low-frequency components is needed better than -20 dB.

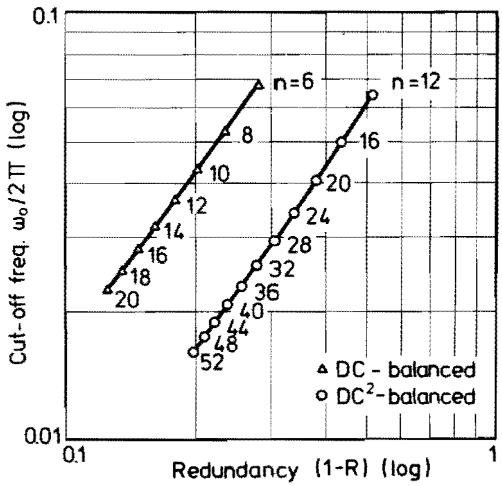

Fig. 2. The cut-off frequency of DC- and DC^2-balanced, zero-disparity codes versus the redundancy $1 - R$. The codeword length is denoted by n.

6. State-dependent encoding

In the case of state-independent encoding the sequence is encoded without information of the past and we have a unique one-to-one mapping of the source words and their channel representations. A larger rate with given codeword length is possible if codewords are allowed to start (and end) in a set of predefined states, called principal states. Franaszek [13,14]) has developed a procedure for determining the existence of such a set of principal states. We found during experiments with this procedure that the following choice of the codeword subsets is satisfactory. The encoder operates on the basic principle of the *bi*-mode alternate coding described by Cattermole[2]). All codewords are chosen in such a way that $d_z = 0$. Codewords with positive and negative d_y are used in two modes, codewords with zero d_y are used without modification in both modes. The choice of the polarity of the codeword is chosen by the encoder in such a way that the RDSS at the end of the new codeword is as close to zero as possible. We can easily verify that the number of encoder states, i.e. the number of RDSS values at the end of a codeword, is twice the number of codeword subsets with different positive d_y values. By properly choosing the initial RDSS value the RDSS at the end of the codewords will assume the values $\pm 1, \pm 3, \ldots, \pm (r-1)$. In practice this is achieved by initializing the RDSS counter with the disparity 1. Using the recursion relations given by eq. (8) we calculated the rate of the *bi*-mode code versus the number of subsets and the length of the codeword. In table II we collected the results using r encoder states.

TABLE II

Code rate R of DC^2-balanced codes with codeword length n and r encoder states

$r =$	2	4	6	8	10
$n =$ 4	.396	.5	—	—	—
8	.488	.557	.594	.625	.641
12	.568	.616	.648	.672	.689
16	.627	.663	.688	.707	.722
20	.670	.700	.720	.736	.748

Note the improvement, for fixed codeword length, in rate with respect to state-independent encoding (table I).

Example 2

An attractive and simple code found in the table has the following parameters: $n = 4$, $r = 4$ and $R = \frac{1}{2}$. Table III shows the codebook of this simple code.

The codewords should be chosen from the table in such a way that the RDSS after transmission of the new codeword is as close to zero as possible. The RDSS assumes the values ± 1 and ± 3; table III shows the corresponding codeword for the given RDSS value. Fig. 3 shows the power spectral density function of this DC^2-balanced code. The spectrum of this code has been calculated using the matrix method of Cariolaro et al.[15,16]). As a reference we plotted the spectrum of the well-known '*bi*-phase' code (zero-disparity,

TABLE III

Codebook of $R = \frac{1}{2}$ DC²-balanced code, the 0's correspond to the -1's of the analysis

source	codeword RDSS = $-3, -1$	d_y	codeword RDSS = $1, 3$	d_y
00	1001	0	1001	0
01	0110	0	0110	0
10	1010	2	0101	-2
11	1100	4	0011	-4

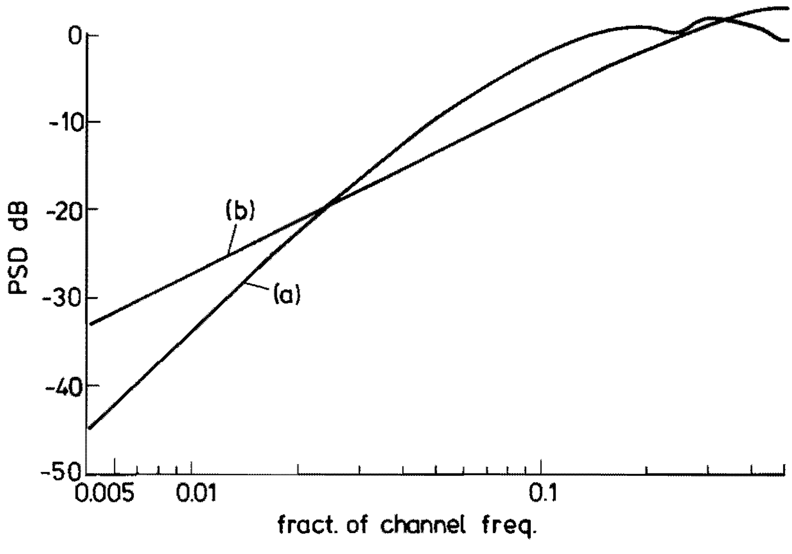

Fig. 3. Power spectral density functions of a DC²-balanced code $n = 4$ and $R = \frac{1}{2}$ (*a*) and '*bi-phase*' (*b*).

DC-balanced code with $n = 2$) having the same rate. Note the improvement in the low-frequency range of the new code.

The memory requirements of Cariolaro's method become prohibitive for a large number of encoder states and large codeword lengths, so that we were not able to compute the spectra of the codes with rate $R > 0.6$.

7. Conclusions

A new class of DC-constrained channel codes, called DC²-balanced codes, have been presented. The power spectral density function $H(\omega)$ of DC²-

balanced codes has besides zero power at zero frequency a zero second-derivative at zero frequency, i.e. $H(0) = H''(0) = 0$.

It was shown that if the redundancy is fixed the cut-off frequency of DC^2-balanced codes is approximately a factor of 2.5 smaller than of classical DC-balanced codes.

Below the cut-off frequencies the spectra of DC^2-balanced codes decreases faster with decreasing frequency than those of DC-balanced codes. We noticed when comparing the spectra of DC^2- and DC-balanced codes that for fixed redundancy a cross-over is found at approximately -20 dB. We conclude that DC^2-balanced codes are superior with respect to DC-balanced codes if rejection of low-frequency components is needed better than -20 dB.

REFERENCES

[1]) J. Justesen, IEEE Trans. Inform. Theory **IT-28**, 457 (1982).
[2]) K. W. Cattermole, Int. J. Electron. **55**, 3 (1983).
[3]) J. M. Griffiths, Electron. Lett. **5**, 79 (1969).
[4]) K. A. Schouhamer Immink, Philips J. Res. **40**, 1 (1985).
[5]) W. R. Bennett, Bell Syst. Techn. J. **37**, 1501 (1958).
[6]) H. Yasuda and H. Inose, Journal IECE Japan **54a**, 506 (1971).
[7]) G. F. M. Beenker, Philips Techn. Note T.N. 105/84, 1984.
[8]) J. Riordan, An introduction to combinatorial analysis, Princeton University Press, 1980.
[9]) T. M. Cover, IEEE Trans. Inform. Theory **IT-19**, 73 (1973).
[10]) R. F. Lyon, IEEE Trans. Commun. **COM-21**, 1438 (1973).
[11]) B. S. Bosik, Bell Syst. Tech. J. **51**, 921 (1972).
[12]) J. N. Frankling and J. R. Pierce, IEEE Trans. Commun. **COM-20**, 1182 (1972).
[13]) P. A. Franaszek, Bell Syst. Tech. J. **47**, 143 (1968).
[14]) P. A. Franaszek, IBM J. Res. Develop. **14**, 376 (1970).
[15]) G. L. Cariolaro and G. P. Tronca, IEEE Trans. Commun. **COM-22**, 1555 (1974).
[16]) G. L. Cariolaro, G. L. Pierobon and G. P. Tronca, Int. J. Electron. **55**, 35 (1983).