Transactions Letters

Discrete Cosine Transform in Error Control Coding

Ja-Ling Wu and Jiun Shiu

Abstract— **We will define a new class of real-number linear block codes using the discrete cosine transform (DCT). We will also show that a subclass with a BCH-like structure can be defined and, therefore, encoding/decoding algorithms for BCH codes can be applied. A (16, 10) DCT code is given as an example.**

I. Introduction

As RECOGNIZED by several authors [1]–[4], error control codes (ECCs) defined over a real or complex field could be advantageous in some aspects including 1) using real/complex operations which are widely available in standard programmable digital signal processors, 2) not restricting to certain block lengths, 3) defining codes which can simultaneously correct errors and reduce data rate, and 4) making error control coding more accessible to signal processing engineers. However, there are problems for a real-number code which are not presented in a finite field. For example, in addition to some impulse errors, the elements of a received code vector will also be contaminated by some unavoidable minor errors due to round-off. This background noise is uncorrectable and will affect decoding algorithms originally designed for codes over a finite field, or new decoding algorithms should be devised specially for codes over a real or complex field. As one of these works, Marshall defined some real-number codes including those based on the discrete Fourier transform (DFT) and the Hadamard transform, and had shown that a DFT code can be decoded by a modified error trapping decoder which is originally used for cyclic code over a finite field [1].

As suggested by Marshall [1], an $(N,\ K)$ real-number block code based on a unitary transform can be defined as follows. First, a generating matrix, $\boldsymbol{G}$ is formed by selecting K rows from the unitary transform. The codeword of $\boldsymbol{y}$ of a K-tuple information vector $\boldsymbol{x}$ is calculated as

$$\boldsymbol{y} = \boldsymbol{x} \cdot \boldsymbol{G}.$$

The codeword will then be transmitted over a channel and received as another N-tuple, say $\boldsymbol{r}$. The received vector $\boldsymbol{r}$ is related to the codeword $\boldsymbol{y}$ as

$$\boldsymbol{r} = \boldsymbol{y} + \boldsymbol{q} + \boldsymbol{e},$$

where $\boldsymbol{q}$ is the minor error vector due to the background noise, and $\boldsymbol{e}$ is the impulse error vector due to the channel noise. A decoder is devised at the receiving side to estimate $\boldsymbol{e}$ based on the redundancy carried in $\boldsymbol{r}$. This estimation procedure will be perturbed by the presence of the background noise.

As long as the background noise is small compared to the information symbols and/or the channel noise, a decoding algorithm originally designed for a code over a finite field can be modified by simply setting a threshold on the accuracy of identifying syndrome patterns. The modified decoding algorithm is then applied to a real-number code in the hope that it will be stable to those small perturbations. The error trapping decoder discussed in [1] serves as an example of this idea. Another example, which is a modified decoding algorithm for real-number BCH codes, will be presented in this paper. Results from numerical evaluations in [1], [3] and here show that there is a good chance for these algorithms to be stable.

The use of the discrete cosine transform (DCT) for defining a class of real-number block codes is presented. As a result not presented here, a nontrivial cyclic subclass can not be defined because that the DCT does not possess the DFT-like translation property. However, it will be shown in this paper that despite the non-cyclic nature of the DCT codes, a set of modified syndromes can be defined with which a modified BCH decoding algorithm can be performed.

This paper consists of three sections. The general definition of the DCT codes is given in Section II. In Section III subclass of these linear block codes with BCH-like structure is defined which can be decoded by some modified decoding algorithms originally designed for BCH codes over a finite field. Finally a (16, 10) DCT code which can correct 3 or fewer errors is presented in Section IV as example.

II. DCT Linear Codes

As defined in [5], the DCT of a data sequence $\boldsymbol{x} = [x_0, x_1, \cdots, x_{k-1}]$ is

$$X_k = \sum_{n=0}^{N-1} x_n \cdot T_k(n) \qquad k = 0, 1, 2, \cdots, N-1, \quad (1)$$

where

$$T_k(n) = \begin{cases} \sqrt{\frac{1}{N}} & k = 0, \\ \sqrt{\frac{2}{N}} \cos \frac{(2n+1)k\pi}{2N} & k = 1, 2, \cdots, N-1 \end{cases}. \quad (2)$$

Paper approved by S. G. Wilson, the Editor for Coding Theory and Applications of the IEEE Communications Society. Manuscript received May 2, 1989; revised May 17, 1993.

The authors are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, 10764, Taiwan, Republic of China.

IEEE Log Number 9406287.

Theorem: (The orthogonal property of the DCT)

$$\sum_{n=0}^{N-1} T_p(n) \cdot T_q(n) = \begin{cases} 1 & p = q \\ 0 & p \neq q \end{cases}. \tag{3}$$

The N equations of (1) can be written in matrix form, using the N-point DCT matrix:

$$\begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_{N-1} \end{bmatrix} = \begin{bmatrix} T_0(0) & T_0(1) & \cdots & T_0(N-1) \\ T_1(0) & T_1(1) & \cdots & T_1(N-1) \\ \vdots & \vdots & & \vdots \\ T_{N-1}(0) & T_{N-1}(1) & \cdots & T_{N-1}(N-1) \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix}. \tag{4}$$

Select any K rows of the N-point DCT matrix, say $j_0, j_1, \cdots, j_{K-1}$, as the rows of a $K \times N$ matrix $\boldsymbol{G}$. Since $\boldsymbol{G}$ is a matrix of rank $\boldsymbol{K}$, it will generate an (N, K) linear block code [10]. The remaining $(N - K)$ rows, say rows $j_K, j_{K+1}, \cdots, j_{N-1}$ which are called parities, form a $(NK) \times K$ matrix $\boldsymbol{H}$. By (3), it is easy to verify that

$$\begin{aligned} \boldsymbol{G} \cdot \boldsymbol{H}^T &= \boldsymbol{0} \\ \boldsymbol{G} \cdot \boldsymbol{G}^T &= \boldsymbol{I}_K. \end{aligned} \tag{5}$$

Thus, by the definition of linear block codes, one can recognize that matrix $\boldsymbol{H}$ is the parity check matrix of the code, and matrix $\boldsymbol{G}^T$ is the right inverse of the generator matrix $\boldsymbol{G}$.

III. A BCH-Like Subclass of DCT Linear Codes

Now we define a subclass of the above DCT linear codes and show how to decode with existing BCH decoding algorithms. It should be noticed that this BCH-like subclass of codes is defined directly from the class of the DCT linear codes, but not conventionally from a cyclic subclass of them.

Suppose that the first d rows, i.e., $0, 1, \cdots, d-1$, are selected as parities, that is

$$\boldsymbol{H} = \begin{bmatrix} T_0(0) & T_0(1) & \cdots & T_0(N-1) \\ T_1(0) & T_1(1) & \cdots & T_1(N-1) \\ \vdots & \vdots & & \vdots \\ T_{d-1}(0) & T_{d-1}(1) & \cdots & T_{d-1}(N-1) \end{bmatrix}_{d \times N}$$

$$\boldsymbol{G} = \begin{bmatrix} T_d(0) & T_d(1) & \cdots & T_d(N-1) \\ T_{d+1}(0) & T_{d+1}(1) & \cdots & T_{d+1}(N-1) \\ \vdots & \vdots & & \vdots \\ T_{N-1}(0) & T_{N-1}(1) & \cdots & T_{N-1}(N-1) \end{bmatrix}_{(N-d) \times N}. \tag{6}$$

The syndrome vector $\boldsymbol{S}$ for a received vector $\boldsymbol{r}$ can be calculated as

$$\begin{aligned} \boldsymbol{S} &= \boldsymbol{r} \cdot \boldsymbol{H} \\ &= (\boldsymbol{q} + \boldsymbol{e}) \cdot \boldsymbol{H} \\ &\doteq \boldsymbol{e} \cdot \boldsymbol{H}, \end{aligned} \tag{7}$$

where we assume that each element of $\boldsymbol{q}$ is much smaller as compared to that of $\boldsymbol{e}$. By moving the normalization factor of $T_i(\cdot)$ to left hand side, the entries of $\boldsymbol{S}$ become

$$S_i' = \frac{1}{f_i} \cdot S_i = \sum_{r=0}^{N-1} e_r \cdot \cos\frac{(2r+1)i\pi}{2N} \quad \text{for } i = 0, 1, 2, \cdots, d-1, \tag{8}$$

where $f_i = \sqrt{1/N}$ for $i = 0$, and $\sqrt{2/N}$ for $i = 1, 2, \cdots, N-1$. This equation can be written as $S' = \boldsymbol{S} \cdot \boldsymbol{P}$ where $\boldsymbol{P}$ is a diagonal matrix with f_i^{-1} as its main diagonal. Before going on, a lemma will be stated. The proof of this lemma can be found in [9].

Lemma:

$$\cos k\omega = \sum_{n=0}^{k} C_{k,n} \cdot (\cos\omega)^n \tag{9}$$

where

$$C_{k,n} = \begin{cases} 0 & k-n \text{ odd} \\ \displaystyle\sum_{m=0}^{\lfloor \frac{n}{2} \rfloor} (-1)^{(k-n)/2} \binom{k}{n-2m} \binom{\frac{k-n}{2}+m}{m} & k-n \text{ even.} \end{cases}$$

Assume that v errors occur, or equivalently, there are only v nonzero terms in the impulse error vector $\boldsymbol{e}$. Suppose that $r_1, r_2, \cdots, r_v$ be the locations of these nonzero terms. Then, by the above lemma, (8) becomes

$$\begin{aligned} S_i' &= \sum_{l=1}^{v} e_{r_l} \sum_{n=0}^{i} C_{i,n} \cdot \left(\cos\frac{(2r_l+1)i\pi}{2N}\right)^n \\ &= [Y_1 \quad Y_2 \quad \cdots \quad Y_v] \begin{bmatrix} 1 & X_1 & X_1^2 & \cdots & X_1^i \\ 1 & X_2 & X_2^2 & \cdots & X_2^i \\ \vdots & & & & \vdots \\ 1 & X_v & X_v^2 & \cdots & X_v^i \end{bmatrix} \cdot \begin{bmatrix} C_{i,0} \\ C_{i,1} \\ \vdots \\ C_{i,1} \end{bmatrix}, \end{aligned} \tag{10}$$

where, for $l = 1, 2, \cdots, v, Y_l = e_{r_l}$ is the error magnitude at location r_l, and $X_l = \cos(2r_l + 1)\pi/2N$ is the error location number for r_l. Writing the $N - K$ syndromes as a vector, we have

$$S' = Y \begin{bmatrix} 1 & X_1 & X_1^2 & \cdots & X_1^{d-1} \\ 1 & X_2 & X_2^2 & \cdots & X_2^{d-1} \\ \vdots & & & & \vdots \\ 1 & X_v & X_v^2 & \cdots & X_v^{d-1} \end{bmatrix} \cdot \begin{bmatrix} C_{0,0} & 0 & C_{2,0} & 0 & \cdots & \cdot \\ 0 & C_{1,1} & 0 & C_{3,1} & \cdots & \cdot \\ 0 & 0 & C_{2,2} & 0 & \cdots & \cdot \\ 0 & 0 & 0 & C_{3,3} & \cdots & \cdot \\ & & \vdots & & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & C_{d-1,d-1} \end{bmatrix} \tag{11}$$

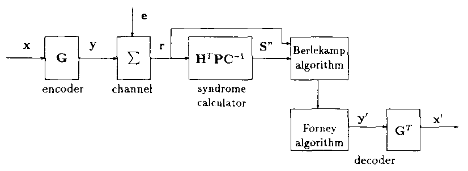

Fig. 1. A (16, 10) DCT code with fast decoding algorithm.

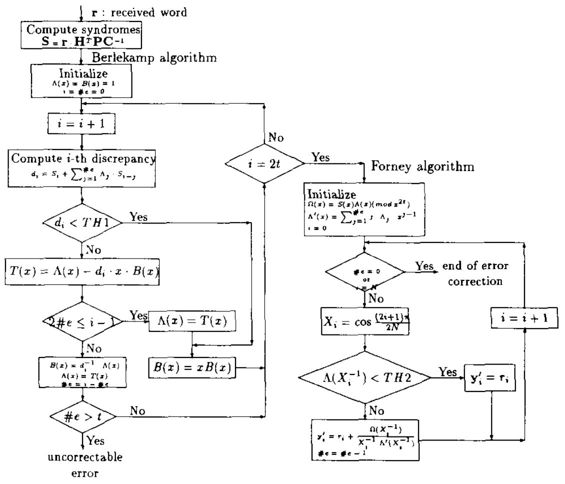
r : received word
Compute syndromes S = r H^TPC^{-1}
Berlekamp algorithm
Initialize Λ(x) = B(x) = 1 i = #e = 0
i = i + 1
Compute i-th discrepancy d_i = S_i + Σ_{j=1}^{#e} Λ_j · S_{i-j}
d_i < TH1
Yes
No
T(x) = Λ(x) − d_i · x · B(x)
2#e ≤ i − 1
Yes
Λ(x) = T(x)
No
B(x) = d_i^{-1} Λ(x) Λ(x) = T(x) #e = i − #e
B(x) = xB(x)
#e > t
No
Yes
uncorrectable error
i = 2t
No
Yes
Forney algorithm
Initialize Ω(x) = S(x)Λ(x)(mod x^{2t}) Λ'(x) = Σ_{j=1}^{#e} j Λ_j x^{j−1} i = 0
#e = 0 or = N
Yes
end of error correction
No
X_i = cos (2i+1)π/2N
i = i + 1
Λ(X_i^{-1}) < TH2
Yes
y'_i = r_i
No
y'_i = r_i + Ω(X_i^{-1}) / X_i^{-1} Λ'(X_i^{-1}) #e = #e − 1

Fig. 2. Berlekamp–Massey and Forney algorithms.

where $\boldsymbol{Y} = [Y_1 \quad Y_2 \quad \cdots \quad Y_v]$. Denote the $d \times d$ upper triangular matrix in (11) by C, which is always invertible. Then

$$S'' = S'C^{-1} = \boldsymbol{Y} \begin{bmatrix} 1 & X_1 & X^{12} & \cdots & X_1^{d-1} \\ 1 & X_2 & X_2^2 & \cdots & X_2^{d-1} \\ \vdots & & & & \vdots \\ 1 & X_v & X_v^2 & \cdots & X_v^{d-1} \end{bmatrix}. \tag{12}$$

By the Peterson–Gorenstein–Zierier algorithm described in [10], it follows that X_i, Y_i, $1 \le i \le v$ can be found by the algorithm provided that

$$v \le \left\lfloor \frac{d}{2} \right\rfloor = t \tag{13}$$

or, in other words, the correction capacity of this code is t. This is also implies that, the DCT codes are maximum distance separable (MDS) under the Hamming distance measure.

As shown in Fig. 1, by introducing the Berlekamp–Massey algorithm [10] and the Forney algorithm [10] with minor modifications, one can have fast decoding algorithm for this subclass of DCT codes. As shown in Fig. 2, a threshold TH1 should be set for testing the ith discrepancy d_i which is the difference between the ith output of the $(i-1)$th autoregressive filter $\Lambda(x)$ and the ith syndrome S_i. This threshold should be small enough to get a better autoregressive model and should be large enough to cope with the background noise. A search loop is included in the part of Forney algorithm in Fig. 2, to find out all legal roots of the error locator polynomial. A legal root is one of the N error locator numbers defined in (10). By this way, a complex procedure for solving the roots of the error locator polynomial is avoided, and the illegal roots due to background noise or an improper value of TH1 are filtered out. The threshold TH2 is set for the accuracy of the evaluation of the error locator polynomial. TH2 should be set large enough to cope with the inaccuracy of the previously synthesized autoregressive filter and small enough to avoid illegal roots.

IV. An Example of DCT Codes

Now a (16, 10) DCT code is constructed as an example which can correct all patterns of 3 or fewer errors. The generator matrix $\boldsymbol{G}$ consists of the last 10 rows of the 16-point DCT matrix, and the parity check matrix $\boldsymbol{H}$ is formed by the other 6 rows. The matrix $\boldsymbol{P}$ is a diagonal matrix with the normalization factors f_i, $i = 0, 1, \cdots, 5$ as its nonzero diagonal elements. The triangular matrix C is as follows:

$$\begin{bmatrix} 1, & 0, & -1, & 0, & 1, & 0 \\ 0, & 1, & 0, & -3, & 0, & 5 \\ 0, & 0, & 2, & 0, & -8, & 0 \\ 0, & 0, & 0, & 4, & 0, & -20 \\ 0, & 0, & 0, & 0, & 8, & 0 \\ 0, & 0, & 0, & 0, & 0, & 16 \end{bmatrix}.$$

Thus, matrix $\boldsymbol{H}^T\boldsymbol{PC}^{-1}$ used for calculating syndromes is shown on the top of the next page.

Some simulation results are given in Table I. As one can see, the arbitrary error patterns with 3 or fewer nonzero terms can be estimated and thus corrected, though coupled with some small background noise. As shown in Table I, the last block is corrupted by 2 impulse errors while the error locator polynomial is of degree 3. There are an extra illegal root which has been filtered out by the threshold TH2.

V. Comparisons and Discussions

Since the proposed DCT codes have nearly the same parameters as the DFT codes defined in [1], the following question arises naturally: whether the DCT codes have any advantages or disadvantages as compared with the DFT codes. In the following, our views on this question are presented.

- The DFT codes are defined by using consecutively indexed powers of the Nth complex root of unity. To generate codewords with real coefficients, the index set included for the generating matrix must be symmetric about 0 (including 0) or about $N/2$ (and including $N/2$ if N is even). This restriction narrows the range of parameters permitted in the codes slightly. On the other hand, the BCH-like subclass of the DCT codes does not have this restriction.
- It was established in [1] that real-number maximum distance separable DFT codes exists for all choices of parameters. Since there is no specific restrictions in the construction of the DCT codes, it follows that the DCT codes exist for all (N, K) and the codes can detect errors equal in number to the rank of $\boldsymbol{H}$, the maximum permitted by the Singleton bound [10].

$$\begin{bmatrix}
1.0, & 0.99518460, & 0.99039268, & 0.98562348, & 0.98087764, & 0.97615433 \\
1.0, & 0.95694029, & 0.91573477, & 0.87630355, & 0.83857018, & 0.80246162 \\
1.0, & 0.88192135, & 0.77778506, & 0.68594533, & 0.60494965, & 0.53351808 \\
1.0, & 0.77301055, & 0.59754509, & 0.46190876, & 0.35706016, & 0.27601138 \\
1.0, & 0.63439327, & 0.40245491, & 0.25531465, & 0.16196999, & 0.10275258 \\
1.0, & 0.47139668, & 0.22221494, & 0.10475136, & 0.04937952, & 0.02327732 \\
1.0, & 0.29028457, & 0.08426523, & 0.02446079, & 0.00710065, & 0.00206111 \\
1.0, & 0.09801716, & 0.00960732, & 0.00094173, & 0.00009224, & 0.00000909 \\
1.0, & -0.09801716, & 0.00960732, & -0.00094173, & 0.00009224, & -0.00000909 \\
1.0, & -0.29028457, & 0.08426523, & -0.02446079, & 0.00710065, & -0.00206111 \\
1.0, & -0.47139668, & 0.22221494, & -0.10475136, & 0.04937952, & -0.02327732 \\
1.0, & -0.63439327, & 0.40245476, & -0.25531465, & 0.16196984, & -0.10275258 \\
1.0, & -0.77301055, & 0.59754509, & -0.46190876, & 0.35706016, & -0.27601138 \\
1.0, & -0.88192135, & 0.77778506, & -0.68594533, & 0.60494965, & -0.53351808 \\
1.0, & -0.95694029, & 0.91573477, & -0.87630355, & 0.83857018, & -0.80246162 \\
1.0, & -0.99518460, & 0.99039268, & -0.98562348, & 0.98087764, & -0.97615427
\end{bmatrix}.$$

TABLE I
SIMULATION RESULT

x	y	e	r	e'	y'	x'
Block 1 : $\Lambda(x) = 1 - 0.0002950x - 0.7775905x^2$						
0.2000000	0.0542908	-0.0000040	0.0542868	0.0000000	0.0542868	0.1996651
-0.3000000	-0.1446468	-0.0000010	-0.1446478	0.0000000	-0.1446478	-0.2996920
0.1000000	0.1945084	0.0900000	0.2845084	0.0899810	0.1945274	0.0997366
-0.3000000	-0.0681906	-0.0000010	-0.0681916	0.0000000	-0.0681916	-0.2998246
0.2000000	-0.2440337	0.0000100	-0.2440237	0.0000000	-0.2440237	0.1999145
0.3000000	0.3180752	-0.0000020	0.3180732	0.0000000	0.3180732	0.3000022
0.1000000	-0.0751739	0.0000010	-0.0751729	0.0000000	-0.0751729	0.1000589
0.3000000	0.0109734	0.0000010	0.0109744	0.0000000	0.0109744	0.2999180
-0.1000000	-0.1162757	-0.0000100	-0.1162857	0.0000000	-0.1162857	-0.0998918
-0.2000000	0.0868550	0.0000100	0.0868650	0.0000000	0.0868650	-0.2000592
	-0.1328392	-0.0000100	-0.1328492	0.0000000	-0.1328492	
	0.1586038	0.0000100	0.1586138	0.0000000	0.1586138	
	0.1995009	0.0000010	0.1995019	0.0000000	0.1995019	
	-0.3490881	0.3000000	-0.0490881	0.2995555	-0.3486435	
	-0.0976906	-0.0000010	-0.0976916	0.0000000	-0.0976916	
	0.2051310	-0.0006000	0.2045310	0.0000000	0.2045310	
Block 2 : $\Lambda(x) = 1 - 0.8818166x - 0.9905048x^2 + 0.8734865x^3$						
0.2000000	0.0542908	-0.2000000	-0.1457092	-0.1995013	0.0537921	0.1996313
-0.3000000	-0.1446468	-0.0000010	-0.1446478	0.0000000	-0.1446478	-0.3003795
0.1000000	0.1945084	0.9000000	1.0945084	0.8992496	0.1952588	0.0996200
-0.3000000	-0.0681906	-0.0003000	-0.0684906	0.0000000	-0.0684906	-0.3002869
0.2000000	-0.2440337	0.0000300	-0.2440037	0.0000000	-0.2440037	0.1998582
0.3000000	0.3180752	-0.0000200	0.3180552	0.0000000	0.3180552	0.3000400
0.1000000	-0.0751739	0.0000010	-0.0751729	0.0000000	-0.0751729	0.1002085
0.3000000	0.0109734	0.0000010	0.0109744	0.0000000	0.0109744	0.3003047
-0.1000000	-0.1162757	-0.0000010	-0.1162767	0.0000000	-0.1162767	-0.0996824
-0.2000000	0.0868550	0.0000010	0.0868560	0.0000000	0.0868560	-0.1998191
	-0.1328392	-0.0000100	-0.1328492	0.0000000	-0.1328492	
	0.1586038	0.0000100	0.1586138	0.0000000	0.1586138	
	0.1995009	0.0000100	0.1995109	0.0000000	0.1995109	
	-0.3490881	0.0000300	-0.3490581	0.0000000	-0.3490581	
	-0.0976906	-0.0000010	-0.0976916	0.0000000	-0.0976916	
	0.2051310	-0.0600000	0.1451310	-0.0599790	0.2051100	
Block 3 : $\Lambda(x) = 1 - 0.7701839x - 1.2011416x^2 + 0.9723981x^3$						
0.2000000	0.0542908	-0.0400000	0.0142909	-0.0437595	0.0580504	0.2010740
-0.3000000	-0.1446468	-0.0000010	-0.1446478	0.0000000	-0.1446478	-0.2989975
0.1000000	0.1945084	0.9000000	1.0945084	0.8999182	0.1945902	0.1009189
-0.3000000	-0.0681906	-0.0000010	-0.0681916	0.0000000	-0.0681916	-0.2991634
0.2000000	-0.2440337	0.0000030	-0.2440307	0.0000000	-0.2440307	0.2007435
0.3000000	0.3180752	-0.0000020	0.3180732	0.0000000	0.3180732	0.3006435
0.1000000	-0.0751739	0.0000010	-0.0751729	0.0000000	-0.0751729	0.1005340
0.3000000	0.0109734	0.0000010	0.0109744	0.0000000	0.0109744	0.3004157
-0.1000000	-0.1162757	-0.0000010	-0.1162767	0.0000000	-0.1162767	-0.0997131
-0.2000000	0.0868550	0.0000010	0.0868560	0.0000000	0.0868560	-0.1998558
	-0.1328392	-0.0000010	-0.1328402	0.0000000	-0.1328402	
	0.1586038	0.0000010	0.1586048	0.0000000	0.1586048	
	0.1995009	0.0000010	0.1995019	0.0000000	0.1995019	
	-0.3490881	0.0000030	-0.3490851	0.0000000	-0.3490851	
	-0.0976906	-0.0000010	-0.0976916	0.0000000	-0.0976916	
	0.2051310	-0.0000060	0.2051250	0.0000000	0.2051250	

- Since only real arithmetic is involved for the computation of the DCT codes, it implies not only the lower computation complexity but also less computation error. As shown in [12, Section 9.8], the output error of a direct computation of the DFT using fixed-point b-bit arithemetic is $(N/32)^{-2b}$ where N is the length of the transform. Under similar conditions, the output error of the DCT is $((N-1)/4N + O(1/N))2^{-2b}$ [13]. In other words, under comparable computation precision, the DCT codes can be decoded more accurately since they suffer less from background noise, mainly due to the computation error.

From the above discussions, it is apparent that the DCT codes do not have some advantages over the DFT codes. Moreover, we would like to point out, in the following, our motivation for this study.

In transform coding applications, there are many consecutive zeros above the cutoff/threshold frequency of a signal, due to the energy packing property of orthogonal discrete transforms [6], [11]. Wolf [3] suggested that these zeros can be viewed as redundancy in the signal and used for cancelling impulse noise. Since the DCT leads to the best performance (in transform coding) among the known fast-computable transforms [11], it is our belief that the DCT provides a good opportunity and as a useful tool for unifying the problems of source coding and channel coding.

VI. CONCLUSION

We have defined a new class of real-number linear codes and shown that a subclass of them has similar structure to the BCH codes defined over a finite field. The BCH bound on code capacity and conventional BCH decoding algorithms can be applied to this new subclass. However, there is no way to describe a nontrivial cyclic subclass of this linear code (a result not shown in this work). This result contrasts the well-known concept: BCH codes are a subclass of cyclic codes. This is simply a difference between codes over the two finite fields and the real field.

REFERENCES

[1] T. G. Marshall, Jr., "Coding of real-number sequences for error correction: A digital signal processing problem," *IEEE Trans. Select. Areas Commun.*, vol. SAC-2, Mar. 1984.

[2] ——, "Codes for error correction based upon interpolation of real-number sequences," in *Proc. 19th Asilomar Conf. Circ. and Syst.*, Pacific Grove, CA, Nov. 1985.

[3] J. K. Wolf, "Redundancy, the discrete transform, and impulse noise cancellation," *IEEE Trans. Commun.*, vol. COM–31, Mar. 1983.

[4] R. Kumaresan, "Rank reduction techniques and burst error-correction decoding in real/complex fields," in *Proc. 19th Asilomar Conf. Circ. Syst.*, Pacific Grove, CA, Nov. 1985.

[5] N. Ahmed, T. Natarjan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C–23, Jan. 1974.

[6] N. S. Jayant and P. Noll, Digital *Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[7] R. Zelinski and P. Noll, "Adapative transform coding of speech signals," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP–25, Aug. 1977.

[8] C. J. Anfinson and F. T. Luk, "A linear algebraic model of algorithm-based fault tolerance," *IEEE Trans. Comput.,* vol. 37, Dec. 1988.

[9] J. H. McClellan and T. W. Parks, "Eigenvalue and eigenvector decomposition of the discrete Fourier transform," *IEEE Trans. Audio and Elect.,* vol. AE–20, Mar. 1972.

[10] R. E. Blahut, *Theory and Practice of Error Control Codes.* Reading, MA: Addison-Wesley, 1983.

[11] R. J. Clark, *Transform Coding of Images.* London, England: Academic, 1985.

[12] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[13] I. D. Yun and S. U. Lee, "On the fixed-point-error analysis of several fast DCT algorithms," *IEEE Trans. Circ. Syst. Video Tech.,* vol. 3, Feb. 1993.