IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received July 16, 2019, accepted August 5, 2019, date of publication August 13, 2019, date of current version August 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934847

On Redundancy Reduction of Non-Recursive Second-Order Spectral-Null Codes

D. PELUSI, (Member, IEEE)
Faculty of Communication Sciences, University of Teramo, 64100 Teramo, Italy
e-mail: dpelusi@unite.it

This work was supported by the Academic Research Funds of the University of Teramo, Italy.

ABSTRACT The code design problem of non-recursive second-Order Spectral Null (2-OSN) codes is to convert balanced information words into 2-OSN words employing the minimum possible redundancy. Let k be the balanced information word length. If $k \in 2I\!N$ then the 2-OSN coding scheme has length $n = k + r$, with 2-OSN redundancy $r \in 2I\!N$ and $n \in 4I\!N$. Here, we use a scheme with $r = 2\log k + \Theta(\log\log k)$. The challenge is to reduce redundancy even further for any given k. The idea is to exploit the degree of freedom to select from more than one possible 2-OSN encoding of a given balanced information word. To reduce redundancy, empirical results suggest that extra information $\delta_k = 0.5\log k + \Theta(\log\log k)$ is obtained. Thus, the proposed approach would give a smaller redundancy $r' = 1.5\log k + \Theta(\log\log k)$ less than $r = 2\log k + \Theta(\log\log k)$.

INDEX TERMS Balanced codes, high order spectral null codes, Knuth's complementation method, parallel decoding scheme, optical and magnetic recording.

I. INTRODUCTION

The spectral-null codes are an important class of codes applied in recording systems. Such codes have zero power spectral density at specific frequencies. The fields of application of spectral-null codes are on transmission systems over fiber or metallic cable and in storage media such as magnetic or optical recording.

Let $SN(n, q)$ indicate the set of qth-order spectral-null words in ϕ^n, with $\phi = \{-1, +1\}$ a bipolar alphabet. This set is defined as (see [6], [7], [16])

$$SN(n, q) \stackrel{def}{=} \left\{X \in \phi^n | m_i(X) = 0, i \in [0, q-1]\right\}. \quad (1)$$

The quantity $m_i(X) = x_1 1^i + x_2 2^i + \ldots + x_n n^i = \sum_{j=1}^{n} x_j j^i$ is the m_i-weight of the word $X = x_1 x_2 \ldots x_n \in \phi^n$, with sums and products over $I\!R$; $m_i(X)$ is also referred to as the i-th moment of X. All the words $X \in SN(n, q)$ are called qth-Order Spectral-Null (q-OSN) words. When $m_i(X) = 0$, the word X is m_i-balanced. Let C be a binary code. C is a q-OSN$(n, \tilde{k})$ code of length n and with $\tilde{k}$ information bits, if, and only if $C \subseteq SN(n, q)$ and $|C| = 2^{\tilde{k}}$. In the case $q = 1$, the q-OSN$(n, \tilde{k})$ codes coincide with the balanced codes [2]–[4], [6], [8], [10], [13]–[16], [19], [21], [23]–[27], [32]. On the other hand, for $q \geq 2$, the q-order spectral null codes are applied in digital recording and partial-response channels [7], [16]. Considering the q-OSN codes over the binary alphabet $Z\!\!Z_2 = \{0, 1\}$ [21] and replacing the symbols -1 and $+1$ with 0 and 1 respectively, $SN(n, q)$ becomes equivalent to the set $SN'(n, q) \subseteq Z\!\!Z_2^n$

$$SN'(n, q) = \left\{ X \in Z\!\!Z_2^n \,\middle|\, m_i(X) = \sum_{j=1}^{n} x_j j^i = \frac{1}{2}\sum_{j=1}^{n} j^i \right\}$$

for all integer $i \in [0, q-1]$. Note that, the sums and products are over $I\!R$.

In the design of a q-OSN code, the main issue is to change the information words into qth-order spectral null words using the minimum possible redundancy. Tallini and Bose [20] found that, for $q = 2$, the minimum redundancy is $r_{min}(k) = 2\log n - 1.141$. Some efficient 2-OSN codes have been proposed in [9], [11], [16], [20], [22], [30], [31]. Tallini *et al.* [28] designed 2-OSN codes whose scheme is based on the combination between the Knuth's parallel decoding proposal [2], [13], [14], [21] and the random walk method for second-Order Spectral Null codes [9], [20], [30]. This approach gives a novel non-recursive efficient codes design method which makes the cited codes less redundant than other code designs. Moreover, the Knuth's parallel decoding scheme has been also used by Weber and Immink [29] and Swart and Weber [18] to convey extra auxiliary data by exploiting the freedom degree to select from more than one possible balancing indexes of a given

The associate editor coordinating the review of this article and approving it for publication was Rui Wang.

information word. Pelusi *et al.* [14] gave a generalization of Knuth's scheme for obtaining efficient m-ary balanced codes with a parallel decoding scheme. In this scheme, the extra information $\delta_k = 0.5\log_m k + \Theta(\log\log k)$ comes from the choice of balancing index and the unused check symbol contribution. A modification of this scheme has been proposed in [12] with the derivation of an asymptotic amount of auxiliary data for a variable length realization of a ternary code construction modification of the scheme in [14].

In this paper, we propose to convey extra auxiliary data using the freedom of choosing more than one possible balanced encoding of a given 2-OSN information word [28]. In other words, we consider the combination between the Knuth's parallel decoding scheme and the random walk method for second-Order Spectral Null codes. The target is to reduce the redundancy $r = 2\log k + \Theta(\log\log k)$ of the second-order spectral null codes designed in [28].

Section II contains a detailed description of the proposed scheme. The experimental results are discussed in Section III. The concluding remarks are contained in Section IV.

II. PROPOSED SCHEME

The proposed scheme is based on the scheme described in [28]. The 2-OSN information words are already m_0-balanced words. Let $\tilde{k}$ be the maximum number of information bits that can be m_0-balanced into k bits, it follows that

$$\tilde{k} = \log_2 \binom{k}{k/2} \tag{2}$$

with $k \in 2I\!N$. Therefore, starting from an even length information word X that belongs to $SN(k,1)$, the design problem is to convert this word into a word in the set $SN(n,2)$ using

$$SN(n,2) = \left\{ X \in \mathbb{Z}_2^n \;\middle|\; \begin{array}{l} m_0(X) = \sum_{j=1}^{n} x_j = \frac{n}{2} \quad \text{and} \\ m_1(X) = \sum_{j=1}^{n} x_j j = \frac{n(n+1)}{4} \end{array} \right\}$$

where $n = k + r$ and r is the check words length. The conversion is assured by suitable functions from an m_1-balancing functions set and appending appropriate check words. In order to define the m_1-balancing functions, the following quantities

$$S(k,\mu_0) \stackrel{def}{=} \{X \in \mathbb{Z}_2^k : m_0(X) = \mu_0\} \subseteq \mathbb{Z}_2^k$$
$$S(k,\mu_0,\mu_1) \stackrel{def}{=} \{X \in \mathbb{Z}_2^k : m_0(X) = \mu_0, m_1(X) = \mu_1\}$$

are defined. Note that $S(k,\mu_0,\mu_1) \subseteq S(k,\mu_0)$, with $k,\mu_0,\mu_1 \in I\!N$. The sets $S(k,\mu_0)$ and $S(k,\mu_0,\mu_1)$ indicate the set of all k-bit m_0-balanced and m_1-balanced data words, respectively.

Let $\langle\Gamma_h\rangle$ be a function from $S(k,k/2)$ into itself, i.e.

$$\langle\Gamma_h\rangle : S(k,k/2) \to S(k,k/2)$$

with $h \in [0,p-1]$ and $p \in I\!N$. Moreover, let $k,r \in 2I\!N$ be given so that $n \stackrel{def}{=} k + r \in 4I\!N$. We define the set

$$CS \stackrel{def}{=} \{\Gamma_0, \Gamma_1, \ldots, \Gamma_{p-1}\},$$

as the set of $p \in I\!N$ non-empty subsets of the set of all the r-bit m_0-balanced check words $S(r,r/2)$.

The sets $\Gamma_0, \Gamma_1, \ldots, \Gamma_{p-1}$ satisfies the following conditions:

1) The sets Γ_h are pair-wise disjoint; i. e.,

$$\Gamma_i \cap \Gamma_j = \emptyset \iff i \neq j.$$

This feature avoids the ambiguity for recovering $h \in [0,p-1]$.

2) For every m_0-balanced information word $X \in S(k,k/2)$ there exists one $h_{bal} \in [0,p-1]$ and $C_{h_{bal}} \in \Gamma_{h_{bal}}$ such that

$$m_1\left(\langle\Gamma_{h_{bal}}\rangle(X)\ C_{h_{bal}}\right) = \frac{n(n+1)}{4}.$$

The meaning of symbol h_{bal} is of an "m_1-balancing index" of X, with $h_{bal}(X) \subseteq [0,p-1]$ as the set of all possible balancing indices of X. Note that, for all $X \in S(k,k/2)$,

$$\begin{aligned} m_0\left(\langle\Gamma_{h_{bal}}\rangle(X)\ C_{h_{bal}}\right) &= m_0\left(\langle\Gamma_{h_{bal}}\rangle(X)\right) + m_0\left(C_{h_{bal}}\right) \\ &= \frac{k}{2} + \frac{r}{2} = \frac{n}{2}. \end{aligned}$$

3) For all indices $h \in [0,p-1]$, the function $\langle\Gamma_h\rangle$ is one-to-one so that, from h and $Y = \langle\Gamma_h\rangle(X)$ it is possible to unambiguously recover X.

Given a word $X = x_1, x_2, \ldots, x_k \in S(k,k/2)$, the random walk method for 2-OSN codes consists of exchanging adjacent bits with any alteration of m_0-weight and with a variation of m_1-weight of -1, 0 or $+1$. The random walk terminates once that the reverse $X^R \stackrel{def}{=} x_k x_{k-1} \ldots x_1$ is achieved. Formally, given

$$X = x_1 x_2 x_3 \ldots x_i \ldots x_j \ldots x_{k-1} x_k$$

let

$$X^{(i,j)} = x_1 x_2 x_3 \ldots \underline{x_j} \ldots \underline{x_i} \ldots x_{k-1} x_k$$

be the word obtained form X by exchanging the ith bit with the jth bit, and

$$X^R = \underline{\underline{x_k x_{k-1} \ldots x_j \ldots x_i \ldots x_3 x_2 x_1}}$$

be the reverse of X.

Formalizing as in [20], let us consider the sequence of $k(k-1)/2+1$ words,

$$
\begin{aligned}
X^{(0)} &\stackrel{def}{=} X && = X,\\
X^{(1)} &\stackrel{def}{=} \left(X^{(0)}\right)^{(1,2)},\\
X^{(2)} &\stackrel{def}{=} \left(X^{(1)}\right)^{(2,3)},\\
&\vdots\\
X^{(k-1)} &\stackrel{def}{=} \left(X^{(k-2)}\right)^{(k-1,k)},\\
X^{(k)} &\stackrel{def}{=} \left(X^{(k-1)}\right)^{(k-2,k-1)},\\
X^{(k+1)} &\stackrel{def}{=} \left(X^{(k)}\right)^{(k-3,k-2)},\\
&\vdots\\
X^{(2k-3)} &\stackrel{def}{=} \left(X^{(2k-4)}\right)^{(1,2)} && = X^{(1,k)},\\
X^{(2k-2)} &\stackrel{def}{=} \left(X^{(2k-3)}\right)^{(2,3)},\\
X^{(2k-1)} &\stackrel{def}{=} \left(X^{(2k-2)}\right)^{(3,4)},\\
&\vdots\\
X^{(3k-6)} &\stackrel{def}{=} \left(X^{(3k-7)}\right)^{(k-2,k-1)},\\
X^{(3k-5)} &\stackrel{def}{=} \left(X^{(3k-6)}\right)^{(k-3,k-2)},\\
X^{(3k-4)} &\stackrel{def}{=} \left(X^{(3k-5)}\right)^{(k-4,k-3)},\\
&\vdots\\
X^{(4k-10)} &\stackrel{def}{=} \left(X^{(4k-9)}\right)^{(2,3)} && = \left(X^{(1,k)}\right)^{(2,k-1)},\\
&\vdots\\
X^{(k(k-1)/2)} &\stackrel{def}{=} \left(X^{(k(k-1)/2)-1}\right)^{(k/2,k/2+1)} = X^R.
\end{aligned}
$$

For example, if $k=4$ and $X=x_1x_2x_3x_4$ then the sequence has cardinality $k(k-1)/2+1=7$ and

$$
\begin{aligned}
X^{(0)} &= x_1x_2x_3x_4 = X,\\
X^{(1)} &= \underline{x_2x_1}x_3x_4\\
X^{(2)} &= x_2\underline{x_3x_1}x_4\\
X^{(3)} &= x_2x_3\underline{\underline{x_4x_1}}\\
X^{(4)} &= x_2\underline{x_4x_3}\underline{\underline{x_1}}\\
X^{(5)} &= \underline{\underline{x_4x_2}}x_3\underline{\underline{x_1}}\\
X^{(6)} &= \underline{\underline{x_4x_3x_2x_1}} = X^R.
\end{aligned}
$$

Given a set of m_1-balancing functions, each m_0-balanced data word $X \in S(k, k/2)$ is encoded as

$$E_2(X) = \langle \Gamma_{h_{bal}} \rangle (X)\ C_{h_{bal}}$$

where $h_{bal} \in h_{bal}(X)$ is an "m_1-balancing index" of X.

The check word partition CS is defined by the following simple constructive rule.

For $h = 0, 1, 2, \ldots, p-1$, for all integers

$$\mu_1 \in m_1 \left(S(r, r/2) - \bigcup_{j=0}^{h-1} \Gamma_j \right), \tag{3}$$

the set Γ_h contains exactly one check word in $S(r, r/2, \mu_1) - \bigcup_{j=0}^{h-1} \Gamma_j$.
This definition implies that

$$p \le \max_{\mu} |S(r, r/2, \mu)|$$

with $\mu \in I\!N$. For example, if $r=8$ and

$$p = \max_{\mu} |S(8, 4, \mu)| = 8$$

then

$$
\begin{aligned}
S(r=8, r/2=4, \mu_1=10) &= \{11110000\},\\
S(r=8, r/2=4, \mu_1=11) &= \{11101000\},\\
S(r=8, r/2=4, \mu_1=12) &= \{11011000, 11100100\},\\
S(r=8, r/2=4, \mu_1=13) &= \{10111000, 11010100, 11100010\},\\
S(r=8, r/2=4, \mu_1=14) &= \{01111000, 10110100, 11001100,\\
&\quad 11010010, 11100001\},\\
S(r=8, r/2=4, \mu_1=15) &= \{01110100, 10101100, 10110010,\\
&\quad 11001010, 11010001\},\\
S(r=8, r/2=4, \mu_1=16) &= \{01101100, 01110010, 10011100,\\
&\quad 10101010, 10110001, 11000110, 11001001\},\\
S(r=8, r/2=4, \mu_1=17) &= \{01011100, 01101010, 01110001,\\
&\quad 10011010, 10100110, 10101001, 11000101\},\\
S(r=8, r/2=4, \mu_1=18) &= \{00111100, 01011010, 01100110, 01101001,\\
&\quad 10010110, 10011001, 10100101, 11000011\},\\
S(r=8, r/2=4, \mu_1=19) &= \{00111010, 01010110, 01011001,\\
&\quad 01100101, 10001110, 10010101, 10100011\},\\
S(r=8, r/2=4, \mu_1=20) &= \{00110110, 00111001, 01001110,\\
&\quad 01010101, 01100011, 10001101, 10010011\},\\
S(r=8, r/2=4, \mu_1=21) &= \{00101110, 00110101, 01001101,\\
&\quad 01010011, 10001011\},\\
S(r=8, r/2=4, \mu_1=22) &= \{00011110, 00101101, 00110011,\\
&\quad 01001011, 10000111\},\\
S(r=8, r/2=4, \mu_1=23) &= \{00011101, 00101011, 01000111\},\\
S(r=8, r/2=4, \mu_1=24) &= \{00011011, 00100111\},\\
S(r=8, r/2=4, \mu_1=25) &= \{00010111\},\\
S(r=8, r/2=4, \mu_1=26) &= \{00001111\};
\end{aligned}
$$

so that, for instance, $CS = \{\Gamma_0, \Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4, \Gamma_5, \Gamma_6, \Gamma_7\}$, where

$$
\begin{aligned}
\Gamma_0 &= \{11110000, 11101000, 11011000, 10111000, 01111000,\\
&\quad 01110100, 01101100, 01011100, 00111100, 00111010,\\
&\quad 00110110, 00101110, 00011110, 00011101, 00011011,\\
&\quad 00010111, 00001111\},\\
\Gamma_1 &= \{11100100, 11010100, 10110100, 10101100, 01110010,\\
&\quad 01101010, 01011010, 01010110, 00111001,\\
&\quad 00110101, 00101101, 00101011, 00100111\},\\
\Gamma_2 &= \{11100010, 11001100, 10110010, 10011100, 01110001,\\
&\quad 01100110, 01011001, 01001110, 01001101,\\
&\quad 00110011, 01000111\},\\
\Gamma_3 &= \{11010010, 11001010, 10101010, 10011010, 01101001,\\
&\quad 01100101, 01010101, 01010011, 01001011\},\\
\Gamma_4 &= \{11100001, 11010001, 10110001, 10100110, 10010110,\\
&\quad 10001110, 01100011, 10001011, 10000111\},\\
\Gamma_5 &= \{11000110, 10101001, 10011001, 10010101, 10001101\},\\
\Gamma_6 &= \{11001001, 11000101, 10100101, 10100011, 10010011\},\\
\Gamma_7 &= \{11000011\}
\end{aligned}
$$

The information word length k and the check word length r must satisfy the (4) [2], [20], [21]

$$\frac{k(k-1)}{2} \le \binom{r}{r/2}. \tag{4}$$

However, there are cases where

$$\frac{k(k-1)}{2} < \binom{r}{r/2},$$

thus there will be $l = \frac{k(k-1)}{2} - \binom{r}{r/2}$ unused check words. Because the partition CS can be constructed in different ways [28], the idea is a novel construction of Γ's. The proposed construction of Γ's is described in Algorithm 1.

Algorithm 1 Proposed Γ's Construction

```
Input: l,p,|Γ|
Output: |Γ|
begin
    for i = (p − 1) : −1 : 0 do
        while (|Γ_i| > 1 and l > 0) do
            |Γ_i| − −;
            α[i] + +;
            l − −;
            if (|Γ_i|>1 and l>0) then
                |Γ_i| − −;
                β[i] − −;
                l − −;
            end
        end
        if l = 0 then
            break;
        end
    end
end
```

For example, if $k = 12$ and $r = 6$ then (4) gives $66 \leq 70$. This means that there are $l = 70 - 66 = 4$ check words unused. The partition CS is constructed according to Algorithm 1, where the choice is to remove 4 unused check words from the Γ's constructed above. The removal procedure consists of removing the check words starting from the last Γ_i (Γ_7 in our case) and taking off the check words from Γ's until their cardinality is equal to 1, i. e. $|\Gamma_i| = 1$. In this way, for the example above, the check words 11001001, 11000101, 10100011, 10010011 are removed from Γ_6 and so $\Gamma_6 = \{10100101\}$.

Now, we define the m_1-image of the words in Γ_h, $m_1(\Gamma_h)$, as the integer interval

$$m_1(\Gamma_h) = [\alpha_h, \beta_h]$$

where

$$\alpha \overset{def}{=} \min_{C \in S(r,r/2)} m_1(C)$$

and

$$\beta \overset{def}{=} \max_{C \in S(r,r/2)} m_1(C)$$

Thus, in the case $r = 8$

$$\begin{aligned} m_1(\Gamma_0) &= [10, 26], \\ m_1(\Gamma_1) &= [12, 24], \\ m_1(\Gamma_2) &= [13, 23], \\ m_1(\Gamma_3) &= [14, 22], \\ m_1(\Gamma_4) &= [14, 22], \\ m_1(\Gamma_5) &= [16, 20], \\ m_1(\Gamma_6) &= [18, 18], \\ m_1(\Gamma_7) &= [18, 18]; \end{aligned}$$

The cardinalities of Γ's are necessary to define the p natural numbers [28]

$$d_h \overset{def}{=} \begin{cases} 0 & \text{if } h = 0, \\ d_{h-1} + \lfloor |\Gamma_{h-1}|/2 \rfloor + \lceil |\Gamma_h|/2 \rceil & \text{if } h \in [1, p-1] \end{cases} \tag{5}$$

In the example, $|\Gamma_0| = 17$, $|\Gamma_1| = 13$, $|\Gamma_2| = 11$, $|\Gamma_3| = 9$, $|\Gamma_4| = 9$, $|\Gamma_5| = 5$, $|\Gamma_6| = 1$ and $|\Gamma_7| = 1$, therefore $d_0 = 0$, $d_1 = 15$, $d_2 = 27$, $d_3 = 37$, $d_4 = 46$, $d_5 = 53$, $d_6 = 56$ and $d_7 = 57$.

According to Theorem 2 in [28], the following m_1-balancing functions for $k = 12$, $r = 8$ and $p = 8$ are defined:

$$\begin{array}{llll} \langle\Gamma_0\rangle(X) & = X^{(0)} & = & x_1x_2x_3x_4x_5x_6x_7x_8x_9x_{10}x_{11}x_{12} \\ \langle\Gamma_1\rangle(X) & = X^{(15)} & = & x_2x_3x_4x_5x_6x_7\underline{x_{12}x_8}x_9x_{10}x_{11}\underline{\underline{x_1}} \\ \langle\Gamma_2\rangle(X) & = X^{(27)} & = & \underline{\underline{x_{12}}}x_3x_4x_5x_6x_7\underline{x_8x_2}x_9x_{10}x_{11}\underline{\underline{x_1}} \\ \langle\Gamma_3\rangle(X) & = X^{(37)} & = & \underline{\underline{x_{12}}}x_3\underline{x_{11}x_4}x_5x_6x_7x_8x_9x_{10}\underline{\underline{x_1x_2}} \\ \langle\Gamma_4\rangle(X) & = X^{(46)} & = & \underline{\underline{x_{12}x_{11}}}x_4x_5x_6x_7x_8\underline{x_{10}x_9}\underline{\underline{x_3x_2x_1}} \\ \langle\Gamma_5\rangle(X) & = X^{(53)} & = & \underline{\underline{x_{12}x_{11}x_{10}}}x_5\underline{x_6x_4}x_7x_8x_9\underline{\underline{x_3x_2x_1}} \\ \langle\Gamma_6\rangle(X) & = X^{(56)} & = & \underline{\underline{x_{12}x_{11}x_{10}}}x_5x_6x_7x_8\underline{x_9x_4}\underline{\underline{x_3x_2x_1}} \\ \langle\Gamma_7\rangle(X) & = X^{(57)} & = & \underline{\underline{x_{12}x_{11}x_{10}}}x_5x_6x_7\underline{x_9x_8}\underline{\underline{x_4x_3x_2x_1}} \end{array}$$

where the underlined parts represent the permutation of bit position.

The steps of 2-OSN encoding is illustrated in Algorithm 2.

Algorithm 2 Encoding Algorithm

```
   Input: Z ∈ Z_2^k̃
   Output: YC = E_2(E_1(Z)), where Y ∈ Z_2^k and C ∈ Z_2^r
1  begin
2      X = E_1(Z);
3      d_0 = 0;
4      for h = 0 : p − 1 do
5          if h ≠ 0 then
6              d_h = d_{h−1} + ⌊|Γ_{h−1}|/2⌋ + ⌈|Γ_h|/2⌉;
7          end
8          ⟨Γ_h⟩(X) = X^(d_h);
9          w_1 = m_1(⟨Γ_h⟩(X));
10         μ_1 = n(n + 1)/4 − w_1 − kr/2;
11         if m_1(C) = μ_1 then
12             h_bal = h;
13         end
14     end
15     ⟨Γ_{h_bal}(X)⟩ = X^(d_{h_bal});
16     Y = ⟨Γ_{h_bal}(X)⟩;
17     C = C_{h_bal};
18     E_2(X) = YC;
19 end
```

Note that, the word X at row 2 of Algorithm 2 is the m_0-balanced word of length k associated with Z, i.e. $X = E_1(Z)$. In this way, $m_0(X) = k/2$. Moreover, refer to row 11, an m_1-balancing check exists in Γ_h if, and only if, $\mu_1 \in m_1(\Gamma_h) = [\alpha_h, \beta_h]$. The value of h_{bal} is one among all possible balancing indices obtained in row 12 and the corresponding m_1-balancing check is

$$C_{h_{bal}} \stackrel{def}{=} C \in \Gamma_{h_{bal}}.$$

The decoding is described in Algorithm 3.

Algorithm 3 Decoding Algorithm

Input: $Y\,C = E_2(E_1(Z))$, where $Z \in \mathbb{Z}_2^{\tilde{k}}$, $Y \in \mathbb{Z}_2^{k}$ and $C \in \mathbb{Z}_2^{r}$.
Output: $Z = E_1^{-1}(E_2^{-1}(Y\,C))$.
begin
  calculate $h_{bal} \in [0, p-1]$ such that $C \in \Gamma_{h_{bal}}$;
  $X = E_2^{-1}(Y\,C)$;
  $Z = E_1^{-1}(X)$;
end

Here, the idea is to exploit the degree of freedom of selecting from more than one possible balancing index h_{bal} in row 12 of Algorithm 2, to transmit extra auxiliary data. For example, we apply the Algorithms 2 and 3 to the m_0-balanced word $X = 110001100110$, where $k = 12$. Therefore, considering

$$M_1 \stackrel{def}{=} \frac{n(n+1)}{n} - \frac{kr}{2} = 105 - 48 = 57,$$

the Algorithm 2 executes the steps $S1$ and $S2$ in rows 8-13 and 15-18 respectively, as follows.

S1 (for $h = 0$ and $d_0 = 0$) Compute:

$$\begin{aligned}\langle\Gamma_0\rangle(X) &= X^{(0)} = 110001100110 (= X)\\ m_1(X^{(0)}) &= 37\\ \mu_1 &= M_1 - m_1(X^{(0)}) = 57 - 37 = 20\end{aligned}$$

The check word $C = 00110110 \in \Gamma_0$ is such that $m_1(C) = \mu_1 = 20 \in m_1(\Gamma_0) = [10, 26]$. So, $h_{bal} = 0$ is an m_1-balancing index and the corresponding m_1-balancing check is $C_0 = 00110110$.

S1 (for $h = 1$ and $d_1 = 15$) Compute:

$$\begin{aligned}\langle\Gamma_1\rangle(X) &= X^{(15)} = 100011000111\\ m_1(X^{(15)}) &= 45\\ \mu_1 &= M_1 - m_1(X^{(15)}) = 57 - 45 = 12\end{aligned}$$

The check word $C = 11100100 \in \Gamma_1$ is such that $m_1(C) = \mu_1 = 12 \in m_1(\Gamma_1) = [12, 24]$. So, $h_{bal} = 1$ is an m_1-balancing index and the corresponding m_1-balancing check is $C_1 = 11100100$.

S1 (for $h = 2$ and $d_2 = 27$) Compute:

$$\begin{aligned}\langle\Gamma_2\rangle(X) &= X^{(27)} = 000011010111\\ m_1(X^{(27)}) &= 52\\ \mu_1 &= M_1 - m_1(X^{(27)}) = 57 - 52 = 5\end{aligned}$$

We have $\mu_1 = 5 \notin m_1(\Gamma_2) = [13, 23]$, then there is no check word $C \in \Gamma_2$ such that $m_1(C) = \mu_1$.

S1 (for $h = 3$ and $d_3 = 37$) Compute:

$$\begin{aligned}\langle\Gamma_3\rangle(X) &= X^{(37)} = 001001100111\\ m_1(X^{(37)}) &= 49\\ \mu_1 &= M_1 - m_1(X^{(37)}) = 57 - 49 = 8\end{aligned}$$

We have $\mu_1 = 8 \notin m_1(\Gamma_3) = [14, 22]$, then there is no check word $C \in \Gamma_3$ such that $m_1(C) = \mu_1$.

S1 (for $h = 4$ and $d_4 = 46$) Compute:

$$\begin{aligned}\langle\Gamma_4\rangle(X) &= X^{(46)} = 010011010011\\ m_1(X^{(46)}) &= 44\\ \mu_1 &= M_1 - m_1(X^{(46)}) = 57 - 44 = 13\end{aligned}$$

We have $\mu_1 = 13 \notin m_1(\Gamma_4) = [14, 22]$, then there is no check word $C \in \Gamma_4$ such that $m_1(C) = \mu_1$.

S1 (for $h = 5$ and $d_5 = 53$) Compute:

$$\begin{aligned}\langle\Gamma_5\rangle(X) &= X^{(53)} = 011010100011\\ m_1(X^{(53)}) &= 40\\ \mu_1 &= M_1 - m_1(X^{(53)}) = 57 - 40 = 17\end{aligned}$$

The check word $C = 10101001 \in \Gamma_5$ is such that $m_2(C) = \mu_1 = 17 \in m_1(\Gamma_5) = [16, 20]$. So, $h_{bal} = 5$ is an m_1-balancing index and the corresponding m_1-balancing check is $C_5 = 10101001$.

S1 (for $h = 6$ and $d_6 = 56$) Compute:

$$\begin{aligned}\langle\Gamma_6\rangle(X) &= X^{(56)} = 011011000011\\ m_1(X^{(56)}) &= 39\\ \mu_1 &= M_1 - m_1(X^{(56)}) = 57 - 39 = 18\end{aligned}$$

The check word $C = 10100101 \in \Gamma_6$ is such that $m_2(C) = \mu_1 = 18 \in m_1(\Gamma_6) = [18, 18]$. So, $h_{bal} = 6$ is an m_1-balancing index and the corresponding m_1-balancing check is $C_6 = 10100101$.

S1 (for $h = 7$ and $d_7 = 57$) Compute:

$$\begin{aligned}\langle\Gamma_7\rangle(X) &= X^{(57)} = 011011000011\\ m_1(X^{(57)}) &= 39\\ \mu_1 &= M_1 - m_1(X^{(57)}) = 57 - 39 = 18\end{aligned}$$

The check word $C = 11000011 \in \Gamma_7$ is such that $m_2(C) = \mu_1 = 18 \in m_1(\Gamma_7) = [18, 18]$. So, $h_{bal} = 7$ is an m_1-balancing index and the corresponding m_1-balancing check is $C_7 = 11000011$.

Now, execute S_2.

S2: for the m_0-balanced information word $X = 010101110001$, we can choose one of the following m_1-balanced codewords as the encoding of X:

$$\begin{aligned}E_2(X) &= X^{(0)}C_0 = 110001100110\ 00110110\\ E_2(X) &= X^{(15)}C_1 = 100011000111\ 11100100\\ E_2(X) &= X^{(53)}C_5 = 011010100011\ 10101001\\ E_2(X) &= X^{(56)}C_6 = 011011000011\ 10100101\\ E_2(X) &= X^{(57)}C_7 = 011011000011\ 11000011\end{aligned}$$

Note that, there is more than one balancing index for the information word $X = 110001100110$. This fact holds for all the information words in $S(12, 6)$. Therefore, we can exploit the freedom of choice of the balancing indices to convey extra auxiliary data, thereby reducing the overall redundancy.

In the decoding phase, by using Algorithm 3, on receiving for example the 2-OSN word

$$YC = 01101010001110101001$$

the last 8 bits 10101001 represent the check word that allows of identifying the m_1-balancing function $\langle\Gamma_h\rangle$ used in the encoding procedure. Since $C \in \Gamma_5$, the remaining sequence 011010100011 is decoded into the m_0-balanced word

$$E_2^{-1}(YC) = \langle\Gamma_5\rangle^{-1}(Y) = 110001100110.$$

Refer to the complexity of Algorithm 2, note that step at row 2 can be accomplished in space $O(k)$ memory bits and time $O(k \log k)$ bit operations by using any of the methods given in [2], [3], [8], [13], [13], [18], [19], [29]. The step in rows 3-14 can be accomplished in space $O(r^5 + k)$ memory bits and time $O(r^3)$ bit operations (see [28]). Moreover, the step for computing $E_2(X)$ takes time $O(1)$ bit operations. Hence, Algorithm 2 has a space complexity of $O(r^5 + k) = O(k)$ memory bits and a time complexity of $O(r^3 \log r + k \log k) = O(k \log k)$ bit operations.

With regard to the decoding, on receiving a codeword $YC = X^{(d_{h_{bal}})}C$, a table look-up indexed by C can be maintained to compute $i = d_{h_{bal}}$ from C. Once $d_{h_{bal}}$ is known, X can be computed from $(d_{h_{bal}}, Y)$ with the giant-baby step based algorithm in [28]. In this way, $T = O(k \log k)$ bit operations are essentially needed to compute X from $(d_{h_{bal}}, Y)$ and $S = O(k^2 \log k)$ memory bits are essentially needed to compute $d_{h_{bal}} \in [0, k(k-1)/2]$ from $C \in \bigcup_{h=0}^{p-1} \Gamma_h$ with the table look-up.

III. RESULTS AND DISCUSSION

Algorithm 2 is run for computing the amount of information coming from the balancing index choice freedom. The Tables 1-9 show the parameters of the 2-OSN coding scheme. The second column shows the quantity $k_{opt} = \log |SN(n, 2)|$; for $n \geq 18$ the values of the second column are obtained with the approximation $|SN(n, 2)| \approx \left\lfloor (4\sqrt{3}/\pi)2^n/n^2 \right\rfloor$ given in [20], [22], that is $k_{opt} = \log \left\lfloor (4\sqrt{3}/\pi)2^n/n^2 \right\rfloor$. The fifth column contains the values of $\tilde{k}$ which come from relation (2). The quantity δ_k represents the extra auxiliary information which comes from the degree of freedom to select between more than one possible balancing indices of each information word.

Tallini *et al.* [28] proposed a method to design non-recursive efficient 2-OSN codes. If $k \in 2IN$ is the 1-OSN code length, then the second-Order Spectral Null coding scheme has length $n = k + r \in 4IN$ with an extra redundancy, $r \in 2IN$ such that $r = 2 \log k + \Theta(\log \log k)$. Here, to improve the redundancy, the choice freedom among the possible balancing indices of a given information word is proposed.

TABLE 1. Results of the proposed scheme from $n = 4$ to $n = 128$, with $n \in 4IN$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
4	1.000	2	2	1	0.000	1.000	0.000
8	3.000	4	4	2	0.333	2.333	0.667
12	5.858	6	6	4	0.708	4.708	1.149
16	9.039	8	8	6	1.485	7.485	1.554
20	12.412	12	8	9	1.703	10.703	1.708
24	15.899	14	10	11	1.944	12.944	2.955
28	19.464	18	10	15	2.079	17.079	2.385
32	23.087	22	10	19	2.270	21.270	1.816
36	26.753	24	12	21	2.371	23.371	3.382
40	30.497	28	12	25	2.551	27.551	2.946
44	34.222	32	12	29	2.567	31.567	2.655
48	37.971	36	12	33	2.652	35.652	2.319
52	41.740	40	12	37	2.654	39.654	2.086
56	45.526	42	14	38	2.793	40.793	4.733
60	49.327	46	14	42	2.858	44.858	4.469
64	53.141	50	14	46	2.926	48.926	4.215
68	56.966	54	14	50	2.998	52.998	3.968
72	60.801	58	14	54	3.030	57.030	3.772
76	64.645	62	14	58	3.027	61.027	3.618
80	68.497	66	14	62	3.021	65.021	3.476
84	72.356	70	14	66	3.106	69.106	3.250
88	76.222	74	14	70	3.118	73.118	3.104
92	80.094	78	14	74	3.186	77.186	2.907
96	83.971	82	14	78	3.165	81.165	2.806
100	87.853	84	16	80	3.264	83.264	4.589
104	91.740	88	16	84	3.231	87.231	4.509
108	95.631	92	16	88	3.323	91.323	4.308
112	99.526	96	16	92	3.246	95.246	4.280
116	103.425	100	16	96	3.379	99.379	4.046
120	107.327	104	16	100	3.348	103.348	3.980
124	111.233	108	16	104	3.372	107.372	3.861
128	115.141	112	16	108	3.353	111.353	3.788

TABLE 2. Results of the proposed scheme from $n = 132$ to $n = 256$, with $n \in 4IN$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
132	119.052	116	16	112	3.411	115.411	3.641
136	122.966	120	16	116	3.435	119.435	3.531
140	126.882	124	16	120	3.404	123.404	3.478
144	130.801	128	16	124	3.456	127.456	3.345
148	134.722	132	16	128	3.516	131.516	3.207
152	138.645	136	16	132	3.516	135.516	3.129
156	142.570	140	16	136	3.571	139.571	2.999
160	146.497	144	16	140	3.551	143.551	2.946
164	150.426	148	16	144	3.566	147.566	2.860
168	154.356	150	18	146	3.588	149.588	4.769
172	158.288	156	16	152	3.561	155.561	2.727
176	162.222	160	16	156	3.608	159.608	2.614
180	166.157	162	18	158	3.629	161.629	4.528
184	170.094	164	20	159	3.658	162.658	7.435
188	174.032	168	20	163	3.684	166.684	7.348
192	177.971	172	20	167	3.704	170.704	7.267
196	181.912	176	20	171	3.742	174.742	7.170
200	185.853	180	20	175	3.726	178.726	7.127
204	189.796	186	18	181	3.702	184.702	5.094
208	193.740	188	20	183	3.758	186.758	6.982
212	197.685	194	18	189	3.749	192.749	4.936
216	201.631	198	18	193	3.754	196.754	4.877
220	205.578	202	18	197	3.768	200.768	4.811
224	209.526	206	18	201	3.803	204.803	4.723
228	213.475	210	18	205	3.816	208.816	4.659
232	217.425	214	18	209	3.790	212.790	4.635
236	221.376	218	18	213	3.809	216.809	4.566
240	225.327	222	18	217	3.802	220.802	4.525
244	229.280	226	18	221	3.820	224.820	4.460
248	233.233	230	18	225	3.849	228.849	4.383
252	237.186	234	18	229	3.848	232.848	4.339
256	241.141	238	18	233	3.855	236.855	4.286

Algorithm 2 has been run for all $n \in 4Z\!\!Z$ from $n = 4$ to $n = 1024$ (see Tables 1-8) and for $n = 2048, 4096, 8192, 16384, 32768$ (see Table 9). Up to $n = 28$, the computation is on all the information words in $S(k, k/2)$, whereas from $n = 32$ to $n = 1024$, we have run our algorithm on randomized information words. In Table 1, the double line splits these two situations. The values of the quantity $\Delta = \tilde{k}_{opt} - \tilde{k}'$ (with $\tilde{k}' = \tilde{k} + \delta\tilde{k}$) are plotted versus $\log n$ (see Figure 1).

TABLE 3. Results of the proposed scheme from $n = 260$ to $n = 384$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
260	245.096	242	18	237	3.882	240.882	4.214
264	249.052	246	18	241	3.886	244.886	4.166
268	253.009	250	18	245	3.906	248.906	4.103
272	256.966	254	18	249	3.881	252.881	4.085
276	260.924	258	18	253	3.896	256.896	4.028
280	264.882	262	18	257	3.931	260.931	3.952
284	268.841	266	18	261	3.916	264.916	3.926
288	272.801	270	18	265	3.943	268.943	3.858
292	276.761	274	18	269	3.948	272.948	3.814
296	280.722	278	18	273	3.966	276.966	3.756
300	284.683	282	18	277	3.965	280.965	3.719
304	288.645	286	18	281	3.990	284.990	3.656
308	292.607	290	18	285	3.994	288.994	3.614
312	296.570	294	18	289	3.979	292.979	3.591
316	300.533	298	18	293	3.986	296.986	3.547
320	304.497	302	18	297	4.013	301.013	3.484
324	308.461	306	18	301	3.999	304.999	3.463
328	312.426	310	18	305	4.032	309.032	3.394
332	316.391	312	20	307	4.060	311.060	5.331
336	320.356	314	22	309	4.056	313.056	7.301
340	324.322	318	22	313	4.068	317.068	7.254
344	328.288	322	22	317	4.090	321.090	7.198
348	332.255	326	22	321	4.088	325.088	7.167
352	336.222	330	22	325	4.079	329.079	7.143
356	340.190	334	22	329	4.080	333.080	7.110
360	344.157	338	22	333	4.134	337.134	7.023
364	348.125	342	22	337	4.148	341.148	6.978
368	352.094	346	22	341	4.137	345.137	6.957
372	356.063	350	22	345	4.139	349.139	6.923
376	360.032	354	22	349	4.149	353.149	6.883
380	364.001	358	22	353	4.163	357.163	6.838
384	367.971	362	22	357	4.165	361.165	6.806

TABLE 4. Results of the proposed scheme from $n = 388$ to $n = 512$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
388	371.941	366	22	361	4.171	365.171	6.770
392	375.912	370	22	365	4.185	369.185	6.727
396	379.882	374	22	369	4.211	373.211	6.671
400	383.853	378	22	373	4.200	377.200	6.654
404	387.825	382	22	377	4.193	381.193	6.631
408	391.796	386	22	381	4.207	385.207	6.589
412	395.768	390	22	385	4.229	389.229	6.539
416	399.740	394	22	389	4.236	393.236	6.504
420	403.712	398	22	393	4.196	397.196	6.516
424	407.685	402	22	397	4.246	401.246	6.439
428	411.658	406	22	401	4.234	405.234	6.424
432	415.631	410	22	405	4.259	409.259	6.372
436	419.605	414	22	409	4.263	413.263	6.342
440	423.578	418	22	413	4.269	417.269	6.309
444	427.552	422	22	417	4.277	421.277	6.275
448	431.526	426	22	421	4.267	425.267	6.259
452	435.501	430	22	425	4.278	429.278	6.223
456	439.475	434	22	429	4.276	433.276	6.199
460	443.450	438	22	433	4.280	437.280	6.170
464	447.425	442	22	437	4.290	441.290	6.135
468	451.400	446	22	441	4.321	445.321	6.080
472	455.376	450	22	445	4.319	449.319	6.056
476	459.351	454	22	449	4.310	453.310	6.041
480	463.327	458	22	453	4.312	457.312	6.015
484	467.303	462	22	457	4.359	461.359	5.944
488	471.280	466	22	461	4.344	465.344	5.935
492	475.256	470	22	465	4.342	469.342	5.914
496	479.233	474	22	469	4.336	473.336	5.897
500	483.209	478	22	473	4.352	477.352	5.858
504	487.186	482	22	477	4.368	481.368	5.818
508	491.164	486	22	481	4.353	485.353	5.811
512	495.141	492	20	487	4.350	491.350	3.791

TABLE 5. Results of the proposed scheme from $n = 516$ to $n = 640$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
516	499.119	494	22	489	4.357	493.357	5.761
520	503.096	498	22	493	4.361	497.361	5.736
524	507.074	502	22	497	4.386	501.386	5.688
528	511.052	506	22	501	4.380	505.380	5.672
532	515.030	510	22	505	4.392	509.392	5.638
536	519.009	514	22	509	4.390	513.390	5.619
540	522.987	518	22	513	4.402	517.402	5.586
544	526.966	522	22	517	4.415	521.415	5.551
548	530.945	526	22	521	4.400	525.400	5.545
552	534.924	530	22	525	4.403	529.403	5.521
556	538.903	534	22	529	4.414	533.414	5.489
560	542.882	538	22	533	4.423	537.423	5.459
564	546.862	542	22	537	4.424	541.424	5.438
568	550.841	546	22	541	4.435	545.435	5.406
572	554.821	550	22	545	4.421	549.421	5.400
576	558.801	554	22	549	4.426	553.426	5.376
580	562.781	558	22	553	4.447	557.447	5.335
584	566.761	562	22	557	4.467	561.467	5.294
588	570.742	566	22	561	4.455	565.455	5.287
592	574.722	570	22	565	4.465	569.465	5.258
596	578.703	574	22	569	4.442	573.442	5.260
600	582.683	578	22	573	4.471	577.471	5.213
604	586.664	582	22	577	4.479	581.479	5.185
608	590.645	586	22	581	4.482	585.482	5.163
612	594.626	590	22	585	4.493	589.493	5.133
616	598.607	594	22	589	4.490	593.490	5.117
620	602.589	598	22	593	4.470	597.470	5.118
624	606.570	602	22	597	4.493	601.493	5.078
628	610.552	606	22	601	4.501	605.501	5.051
632	614.533	610	22	605	4.518	609.518	5.015
636	618.515	614	22	609	4.501	613.501	5.014
640	622.497	618	22	613	4.514	617.514	4.983

TABLE 6. Results of the proposed scheme from $n = 644$ to $n = 768$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
644	626.479	622	22	617	4.506	621.506	4.973
648	630.461	626	22	621	4.518	625.518	4.943
652	634.444	630	22	625	4.543	629.543	4.900
656	638.426	634	22	629	4.526	633.526	4.900
660	642.408	638	22	633	4.526	637.526	4.882
664	646.391	642	22	637	4.535	641.535	4.856
668	650.374	646	22	641	4.531	645.531	4.843
672	654.356	650	22	645	4.537	649.537	4.820
676	658.339	654	22	648	4.560	652.560	5.779
680	662.322	658	22	652	4.544	656.544	5.778
684	666.305	662	22	656	4.560	660.560	5.745
688	670.288	666	22	660	4.547	664.547	5.741
692	674.272	670	22	664	4.558	668.558	5.714
696	678.255	674	22	668	4.574	672.574	5.681
700	682.239	678	22	672	4.568	676.568	5.671
704	686.222	682	22	676	4.578	680.578	5.644
708	690.206	686	22	680	4.562	684.562	5.643
712	694.190	690	22	684	4.579	688.579	5.610
716	698.173	694	22	688	4.571	692.571	5.603
720	702.157	698	22	692	4.581	696.581	5.576
724	706.141	702	22	696	4.574	700.574	5.568
728	710.125	706	22	700	4.591	704.591	5.535
732	714.110	710	22	704	4.603	708.603	5.506
736	718.094	714	22	708	4.607	712.607	5.487
740	722.078	718	22	712	4.597	716.597	5.481
744	726.063	722	22	716	4.622	720.622	5.441
748	730.047	726	22	720	4.613	724.613	5.434
752	734.032	730	22	724	4.627	728.627	5.405
756	738.017	734	22	728	4.623	732.623	5.393
760	742.001	738	22	732	4.624	736.624	5.377
764	745.986	742	22	736	4.638	740.638	5.348
768	749.971	746	22	740	4.644	744.644	5.327

The vertical line represents the separation between exact and approximated computation. In the first case, we consider the balancing index choice freedom for all the information words, whereas in the second case, the computation is on a set of information words chosen in a random way. In the approximated case, the values of δ_k are computed by taking the average on 1 million samples. The solid line shows the trend of Δ over $\log n$. Observing the graph of Figure 1, we can note that Δ tends to increase with $\log n$. Considering the values of Δ in Table 1 and following the outcomes of [8] and [20], we can define Δ as

$$\Delta = 0.5 \log n + 0.5 \log \log n - 1.467, \qquad (6)$$

because the minimum redundancy of the balanced codes is $1/2 \log n + 0.326$ [8], whereas the minimum redundancy of 2-OSN codes is $2 \log n - 1.141$ [20]. Thus, the total 2-OSN redundancy is $2 \log n - 1.141 - (1/2 \log n + 0.326) = 3/2 \log n - 1.467$.

TABLE 7. Results of the proposed scheme from $n = 772$ to $n = 896$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
772	753.956	750	22	744	4.626	748.626	5.330
776	757.941	754	22	748	4.635	752.635	5.306
780	761.926	758	22	752	4.632	756.632	5.294
784	765.912	762	22	756	4.662	760.662	5.250
788	769.897	766	22	760	4.651	764.651	5.246
792	773.882	770	22	764	4.655	768.655	5.228
796	777.868	774	22	768	4.658	772.658	5.210
800	781.853	778	22	772	4.677	776.677	5.176
804	785.839	782	22	776	4.666	780.666	5.173
808	789.825	786	22	780	4.659	784.659	5.166
812	793.810	790	22	784	4.664	788.664	5.146
816	797.796	794	22	788	4.678	792.678	5.118
820	801.782	798	22	792	4.698	796.698	5.084
824	805.768	802	22	796	4.680	800.680	5.088
828	809.754	806	22	800	4.683	804.683	5.071
832	813.740	810	22	804	4.677	808.677	5.063
836	817.726	814	22	808	4.697	812.697	5.029
840	821.712	818	22	812	4.683	816.683	5.029
844	825.699	822	22	816	4.697	820.697	5.002
848	829.685	826	22	820	4.688	824.688	4.997
852	833.672	830	22	824	4.701	828.701	4.970
856	837.658	834	22	828	4.705	832.705	4.953
860	841.645	838	22	832	4.718	836.718	4.926
864	845.631	842	22	836	4.722	840.722	4.909
868	849.618	846	22	840	4.718	844.718	4.900
872	853.605	850	22	844	4.704	848.704	4.901
876	857.591	854	22	848	4.717	852.717	4.874
880	861.578	858	22	852	4.726	856.726	4.853
884	865.565	862	22	856	4.739	860.739	4.827
888	869.552	866	22	860	4.740	864.740	4.812
892	873.539	870	22	864	4.739	868.739	4.800
896	877.526	874	22	868	4.729	872.729	4.798

TABLE 8. Results of the proposed scheme from $n = 900$ to $n = 2014$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
900	881.513	878	22	872	4.752	876.752	4.761
904	885.501	882	22	876	4.735	880.735	4.765
908	889.488	886	22	880	4.749	884.749	4.739
912	893.475	890	22	884	4.734	888.734	4.742
916	897.463	894	22	888	4.746	892.746	4.716
920	901.450	898	22	892	4.743	896.743	4.707
924	905.437	902	22	896	4.734	900.734	4.703
928	909.425	906	22	900	4.751	904.751	4.674
932	913.413	910	22	904	4.766	908.766	4.646
936	917.400	914	22	908	4.766	912.766	4.634
940	921.388	918	22	912	4.755	916.755	4.632
944	925.376	922	22	916	4.758	920.758	4.618
948	929.363	926	22	920	4.761	924.761	4.602
952	933.351	930	22	924	4.768	928.768	4.584
956	937.339	934	22	928	4.803	932.803	4.536
960	941.327	938	22	932	4.791	936.791	4.536
964	945.315	942	22	936	4.776	940.776	4.539
968	949.303	946	22	940	4.773	944.773	4.531
972	953.291	950	22	944	4.783	948.783	4.508
976	957.280	954	22	948	4.795	952.795	4.485
980	961.268	958	22	952	4.791	956.791	4.476
984	965.256	962	22	956	4.809	960.809	4.447
988	969.244	966	22	960	4.813	964.813	4.431
992	973.233	970	22	964	4.806	968.806	4.427
996	977.221	974	22	968	4.806	972.806	4.415
1000	981.209	978	22	972	4.808	976.808	4.402
1004	985.198	982	22	976	4.803	980.803	4.395
1008	989.186	986	22	980	4.816	984.816	4.370
1012	993.175	990	22	984	4.824	988.824	4.351
1016	997.164	994	22	988	4.836	992.836	4.328
1020	1001.152	998	22	992	4.820	996.820	4.333
1024	1005.141	1002	22	996	4.816	1000.816	4.325

The trend of Δ as defined in (6) is plotted in the graph of Figure 1 (see the dashed line). Note that, the function Δ in (6) well approximates the experimental values of Δ in Table 1. On the other hand, from the data in the table, we conjecture

$$\delta_k = 0.5 \log k + \Theta(\log \log k) \tag{7}$$

and

$$r' = n - \tilde{k}' = 1.5 \log k + \Theta(\log \log k). \tag{8}$$

TABLE 9. Results of the proposed scheme for $n = 2048, 4096, 8192, 16384, 32768$, with $n \in 4\mathbb{N}$.

n	$\tilde{k}_{opt}$	k	$r = n - k$	$\tilde{k}$	$\delta\tilde{k}$	$\tilde{k}' = \tilde{k} + \delta\tilde{k}$	$\Delta = \tilde{k}_{opt} - \tilde{k}'$
2048	2027.141	2024	24	2018	5.187	2023.187	3.954
4096	4073.141	4070	26	4063	5.654	4068.654	4.487
8192	8167.141	8164	28	8157	6.219	8163.219	3.922
16384	16357.141	16354	30	16346	6.699	16352.699	4.442
32768	32739.141	32736	32	32728	7.386	32735.386	3.755

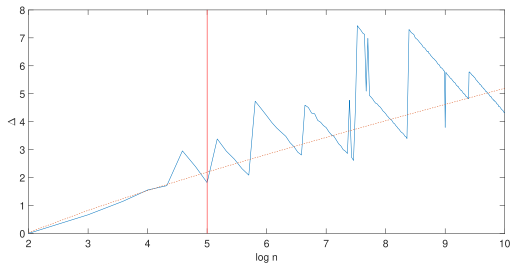

FIGURE 1. Trend of the experimental $\Delta = \tilde{k}_{opt} - \tilde{k}'$ (solid line) and the function $\Delta = 0.5 \log n + 0.5 \log \log n - 1.467$ (dashed line) over $\log n$.

This means that the redundancy is reduced comparing (8) with $r = 2 \log k + \Theta(\log \log k)$ in [28]. In other terms, thanks to the extra information δ_k in (7) , an improvement of the 2-OSN codes designed in [28] is achieved.

IV. CONCLUSION

The idea to exploit the degree of freedom to select between more than one possible balancing encoding of a given information word, was proposed by Weber and Immink [29], Swart and Weber [18], Pelusi *et al.* [14] and Paluncic and Maharaj [12]. Auxiliary data can be used to reduce the redundancy of Knuth's simple balancing method. The proposed approach applies the balancing index choice freedom to the 2-order spectral null codes designed in [28]. In the proposed scheme, the algorithms are run over all the information words for n from 4 to 28 and averaged over 1 million samples from 32 to 1024. The results show that the extra information $\delta_k = 0.5 \log k + \Theta(\log \log k)$ is conveyed from the encoder. This means that the redundancy is equal to $1.5 \log k + \Theta(\log \log k)$. Therefore, the redundancy r is reduced in respect to the scheme proposed in [28], where $r = 2 \log k + \Theta(\log \log k)$. Finally, our scheme improves the 2-OSN codes in [28]. Future research directions will focus on the application of the proposed approach also to q-OSN$(n; \tilde{k})$ codes, with $q > 2$.

ACKNOWLEDGMENT

D. Pelusi thanks Prof. Luca Tallini for the precious suggestions and support.

REFERENCES

[1] N. Alon, E. E. Bergmann, D. Coppersmith, and A. M. Odlyzko, "Balancing sets of vectors," *IEEE Trans. Inf. Theory*, vol. IT-34, no. 1, pp. 128–130, Jan. 1988.

[2] S. Al-Bassam and B. Bose, "On balanced codes," *IEEE Trans. Inf. Theory*, vol. 36, no. 2, pp. 406–408, Mar. 1990.

[3] S. Al-Bassam and B. Bose, "Design of efficient balanced codes," *IEEE Trans. Comput.*, vol. 43, no. 3, pp. 362–365, Mar. 1994.

[4] B. Bose, "On unordered codes," *IEEE Trans. Comput.*, vol. 40, no. 2, pp. 125–131, Feb. 1991.
[5] T. M. Cover, "Enumerative source encoding," *IEEE Trans. Inf. Theory*, vol. IT-19, no. 1, pp. 73–77, Jan. 1973.
[6] K. A. S. Immink, "Spectrum shaping with binary DC_2-constrained channel codes," *Philips J. Res.*, vol. 40, no. 1, pp. 40–53, 1985.
[7] K. A. S. Immink, *Codes for Mass Data Storage Systems*, 2nd ed. Eindhoven, The Netherlands: Shannon Foundation, 2004.
[8] D. Knuth, "Efficient balanced codes," *IEEE Trans. Inf. Theory*, vol. IT-32, no. 1, pp. 51–53, Jan. 1986.
[9] R. Mascella and L. G. Tallini, "On efficient high-order spectral-null codes over the *m*-ary alphabet," *J. Discrete Math. Sci. Cryptogr.*, vol. 8, no. 3, pp. 459–481, Jan. 2005.
[10] R. Mascella and L. G. Tallini, "Efficient *m*-ary balanced codes which are invariant under symbol permutation," *IEEE Trans. Comput.*, vol. 55, no. 8, pp. 929–946, Aug. 2006.
[11] R. Mascella, D. Pelusi, L. Pezza, S. Elmougy, L. G. Tallini, and B. Bose, "On efficient second-order spectral-null codes using sets of m_1-balancing functions," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, Jul. 2013, pp. 141–145.
[12] F. Palunčić and B. T. Maharaj, "Capacity-approaching non-binary balanced codes using auxiliary data," *IEEE Trans. Inf. Theory*, vol. 65, no. 1, pp. 159–173, Jan. 2019. doi: 10.1109/TIT.2018.2844834.
[13] D. Pelusi, L. G. Tallini, and B. Bose, "On *m*-ary balanced codes with parallel decoding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Austin, TX, USA, Jun. 2010, pp. 1305–1309.
[14] D. Pelusi, S. Elmougy, L. G. Tallini, and B. Bose, "*m*-ary balanced codes with parallel decoding," *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3251–3264, Jun. 2015.
[15] L. Pezza, L. G. Tallini, and B. Bose, "Variable length unordered codes," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 548–569, Feb. 2012.
[16] R. M. Roth, P. H. Siegel, and A. Vardy, "High-order spectral-null codes-constructions and bounds," *IEEE Trans. Inf. Theory*, vol. 40, no. 6, pp. 1826–1840, Nov. 1994.
[17] K. A. S. Immink and J. H. Weber, "Very efficient balanced codes," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 2, pp. 188–192, Feb. 2010.
[18] T. G. Swart and J. H. Weber, "Efficient balancing of q-ary sequences with parallel decoding," in *Proc. IEEE Int. Symp. Inf. Theory*, Seoul, South Korea, Jun./Jul. 2009, pp. 1564–1568.
[19] L. G. Tallini, R. M. Capocelli, and B. Bose, "Design of some new efficient balanced codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 790–802, May 1996.
[20] L. G. Tallini and B. Bose, "On efficient high-order spectral-null codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 7, pp. 2594–2601, Nov. 1999.
[21] L. G. Tallini and B. Bose, "Balanced codes with parallel encoding and decoding," *IEEE Trans. Comput.*, vol. 48, no. 8, pp. 794–814, Aug. 1999.
[22] L. Tallini, "Geometric properties of high-order spectral-null codes," *J. Pure Appl. Math.*, vol. 14, pp. 149–176, Oct. 2003.
[23] L. G. Tallini, "Bounds on the capacity of the unidirectional channels," *IEEE Trans. Comput.*, vol. 54, no. 2, pp. 232–235, Feb. 2005.
[24] L. G. Tallini, S. Elmougy, and B. Bose, "Analysis of plain and diversity combining hybrid ARQ protocols over the $m(\geq 2)$-ary asymmetric channel," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5550–5558, Dec. 2006.
[25] L. G. Tallini, S. Al-Bassam, and B. Bose, "Feedback codes achieving the capacity of the Z-channel," *IEEE Trans. Inf. Theory*, vol. 54, no. 3, pp. 1357–1362, Mar. 2008.
[26] L. G. Tallini and U. Vaccaro, "Efficient *m*-ary balanced codes," *Discrete Appl. Math.*, vol. 92, no. 1, pp. 17–56, Mar. 1999.
[27] L. G. Tallini and B. Bose, "On L_1 metric asymmetric/unidirectional error control codes, constrained weight codes and σ-codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, Jul. 2013, pp. 694–698.
[28] L. G. Tallini, D. Pelusi, R. Mascella, L. Pezza, S. Elmougy, and B. Bose, "Efficient non-recursive design of second-order spectral-null codes," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3084–3102, Jun. 2016.
[29] J. H. Weber and K. A. S. Immink, "Knuth's balanced codes revisited," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1673–1679, Apr. 2010.
[30] C.-N. Yang, "Efficient encoding algorithm for second-order spectral-null codes using cyclic bit shift," *IEEE Trans. Comput.*, vol. 57, no. 7, pp. 876–888, Jul. 2008.
[31] C.-N. Yang, Z.-Y. Lin, and S.-L. Peng, "Reducing code length of second-order spectral-null code," *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 492–503, Feb. 2015.
[32] J.-H. Youn and B. Bose, "Efficient encoding and decoding schemes for balanced codes," *IEEE Trans. Comput.*, vol. 52, no. 9, pp. 1229–1232, Sep. 2003.

D. PELUSI (M'17) received the Ph.D. degree in computational astrophysics from the University of Teramo, Italy, where he is currently an Associate Professor with the Faculty of Communication Sciences. His research interests include fuzzy logic, neural networks, information theory, evolutionary algorithms, and machine learning. He has served as a Program Member of many conferences and as an Editorial Board Member of many journals. He is also an Associate Editor of the IEEE Transactions on Emerging Topics in Computational Intelligence, IEEE Access, the *International Journal of Machine Learning and Cybernetics* (Springer), and *Array* (Elsevier). He is also a Guest Editor for Elsevier, Springer, and Inderscience journals. He is also a Reviewer of reputed journals, such as the IEEE Transactions on Fuzzy Systems and the IEEE Transactions on Neural Networks and Learning Systems.