Majority Logic Coding and its Multinomial Representation

John B. Moore and Keng T. Tan

Abstract—

Multinomial representations are derived for majority logic operations on bipolar binary data. The coefficients are given simply in terms of the readily computed lower *Cholesky* factor of *Pascal Matrices* of order n for codes of block length n.

Sufficient conditions on majority logic codes having no deterministic errors introduced in the majority logic coding and majority logic decoding process are given in terms of the multinomial coefficients and codewords. These are used to guarantee that certain majority logic codes based on Hadamard matrices, Rademacher matrices and *pseudo*-random bipolar sequences have no deterministic coding/decoding errors.

Building on this work, a novel majority logic coding scheme is proposed which comprises a first stage of linear coding with its own spreading factor, followed by a second stage of majority logic coding with its own spreading factor, and a majority logic based decoding to recover the data.

I. Introduction

Majority voting on binary data is the basis of certain nonlinear block coding schemes in communication systems. The majority logic operation is used in both the coding and decoding operations. Because of the nonlinearity of the operation, there is difficulty in predicting system performance, or seeing how to improve system performance. A crucial tool in this task is a multinomial representation of the majority logic operation.

A multinomial expansion for majority logic has been partially studied in [1], [2], and the results applied in various communication contexts. General formulas for the first and last coefficients in the expansion are stated, and for bipolar binary vectors of length n, it is claimed that the even numbered coefficients are zero for n even, but we know of no sources which give other coefficients.

Here, we first give a complete theory for the multinomial representations of majority logic operations on bipolar binary data. The majority logic operation can be a classic *sign* function of the sum of the binary data, as studied in the earlier literature known to us. Perhaps more usefully, we also give a theory for what we term here $sign_{\pm}$ functions. These are *sign* operations where an output of 0 is replaced by $+1$ or -1. The approach extends to other nonlinear functions of the sum of binary data, such as to *sigmoidal functions* used in artificial neural networks. It also extends to arbitrary nonlinear functions of bipolar binary data vectors that are invariant of the order of the data within the vector.

The coefficients of the multinomial expansion are linear in what we call a generalized Pascal matrix, which can be factored in terms of the lower triangular *Cholesky* factor, denoted here P_n, of a *Pascal matrix* of order n. The 'new' results are generalizations of the classical results. It would not be surprising if at least some of the results were known by Pascal, but the motivation for deriving them, or highlighting them, is coming from applications of nonlinear coding for next generation wireless communications.

The explicit multinomial representations for majority logic are applied first to majority logic based coding and decoding schemes. Such schemes have been proposed in the literature [3], [4], [5], [6], [7], but theory guaranteeing that proposed code classes lead to error free communication in the noise-free channel case has been limited. Here we develop sufficient conditions based on a diagonal dominance condition of matrices derived from a code matrix and the multinomial coefficients. The condition is applied to codes based on Hadamard and Rademacher matrices, and *pseudo*-random PN sequences, leading to new results in some cases.

A novel majority logic coding scheme is proposed which comprises a first stage of linear coding with its own spreading factor, followed by a second stage of majority logic coding with an additional spreading factor. This doubly coded data is recovered by a majority logic based decoding. The first linear stage can be simply an augmentation of parity bits.

In Section II, the new results on multinomial expansions of majority logic functions are derived. In Section III, these results are applied to majority logic based nonlinear block coding. In Section IV, the estimation of partial products of data are presented, and new coding algorithms based on these are proposed in Section V. A diagonal dominance codeword condition and its application is presented in Section VI. In Section VII, numerical results of the schemes analyzed in this paper are presented. Conclusions are drawn in Section VIII.

II. The Pascal matrix and a multinomial expansion

In this section, we review background material on classical results in order to set up notation for the main results of the following sections.

Our results concern nonlinear operations on a data n-vector $a = [a_1, a_2, \ldots, a_n]'$ with $a_i \in \{+1, -1\}$. Now any nonlinear function of a belongs to a finite discrete set of no more than 2^n elements. Indeed, such functions are linear in an indicator 2^n-vector $Y \in \{e_1, e_2, \ldots, e_n\}$, where e_i is a zero 2^n-vector save that the i^{th} element is unity.

Our new results concern nonlinear operations that are invariant of any ordering in the data, such as functions of $\sum_{i=1}^n a_i$, $\prod_{i=1}^n a_i$,or of $\prod_{i=1}^n (1 + a_i)$. In this case, the functions belong to a discrete set of at most $n + 1$ elements.

Our focus is on (nonlinear) majority logic functions involving *sign* operations on sums of partial products a_i, which map one-to-one to the data vector. The resulting representations are termed multinomial representations.

A. Multinomial representation for majority logic

A.1 Nonlinear functions and majority logic

Consider the *sign* function definition.

$$sign(x) := \begin{cases} 1 \text{ if } x > 0 \\ 0 \text{ if } x = 0 \\ -1 \text{ if } x < 0 \end{cases} . \tag{1}$$

Let us also introduce derivative definitions, denoted $sign_+$ and $sign_-$ as

$$sgn_{\pm}(x) := \begin{cases} 1 \text{ if } x > 0 \\ \pm 1 \text{ if } x = 0 \\ -1 \text{ if } x < 0 \end{cases} . \tag{2}$$

Consider now a set of n bipolar binary digits $\{a_1, a_2, \ldots, a_n\}$, that is where $a_i \in \{+1, -1\}$. The majority logic operation

The Department of Information Engineering, Research School of Information Sciences and Engineering,The Australian National University, Canberra, ACT 0200, Australia, john.moore@anu.edu.au, and a.tan@ecu.edu.au. The work has partial support from the Earmarked RGC grant CUHK 4227/00E and the Australian Research Grants Committee Discovery grant A00105829

on this n-block of data is simply $sign_*(\sum_1^n a_i)$, where we have used $sign_*$ to denote either $sign$, $sign_+$ or $sign_-$. The latter two options can be used if the output of the logic operation is constrained to be also bipolar binary.

A.2 Multinomial representation

Early literature [1][2], presents a multinomial representation for the majority logic $sign$ operation, which we here also mildly generalize as

$$sign_*(\sum_{i=1}^{n} a_i) = \rho_0 + \rho_1 \sum_{i=1}^{n} a_i + \rho_2 \sum_{\text{all } i>j} a_i a_j + \rho_3 \sum_{\text{all } i>j>k} a_i a_j a_k + \cdots + \rho_n \prod_{i=1}^{n} a_i. \quad (3)$$

for suitable selections of coefficients $\rho := [\rho_0, \rho_1, \ldots, \rho_n]'$, which will depend on which of the $sign$ operations is used. The selection of the coefficients and their properties is the study of this paper.

Of course, the nonlinear function $\prod_{i=1}^{n}(1 + a_i)$ has a multinomial expansion given by the right hand side of (3) with coefficients $\rho = [1 \;\; 1 \ldots 1]$. The mapping from the set $\{a_i\}$ to the set of the sums of products in (3) via the coefficients of the ρ_i, is known to be one to one.

The earlier work has given specific formulas for the coefficients ρ_0, ρ_1, ρ_n of (3) in terms of permutation operations ${}^nC_i = \frac{n!}{i!(n-i)!}$, at least for the case of the classic $sign$ function. It is also noted in the early work that in this case $\rho_i = 0$ for n, i even, but other coefficients have not been studied to our knowledge.

In our applications of such expansions, it is important to have readily calculated coefficients for all the coefficients ρ_i, and to see relationships between them in order to understand experimentally observed relationships in majority logic coding/decoding for communication systems.

In order to proceed, we first review relevant results of the Pascal matrix.

B. The lower Cholesky factor of the Pascal matrix

The well known (second) Pascal matrix, is a lower *Cholesky* factor of the original (first) Pascal matrix. We will refer to this (second) Pascal matrix simply as the *Pascal matrix*, and use the notation $P_n = (p_n^{i,j})$ for an $n \times n$ such matrix. Its elements, for $i, j = 1, 2, \ldots, n$ are defined in terms of the binomial coefficients , so that the i, j element for $i \le j$ is

$$p_n^{i,j} = (-1)^{j-1}.{}^{i-1}C_{j-1} := \frac{(-1)^{j-1}(i-1)!}{(j-1)!(i-j)!}, \;\; \text{for} \;\; i \ge j. \quad (4)$$

The key property which we exploit subsequently is that P_n is *involutary* in that

$$P_n = P_n^{-1}, \;\; P_n P_n = I_n. \quad (5)$$

We see that P_n has the form,

$$P_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \cdot & \cdot \\ 1 & -1 & 0 & 0 & 0 & 0 & \cdot & \cdot \\ 1 & -2 & 1 & 0 & 0 & 0 & \cdot & \cdot \\ 1 & -3 & 3 & -1 & 0 & 0 & \cdot & \cdot \\ 1 & -4 & 6 & -4 & 1 & 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ {}^{(n-1)}C_0 & {}^{(n-1)}C_1 & {}^{(n-1)}C_2 & \cdot & \cdot & \cdot & \cdot & {}^{(n-1)}C_n \end{bmatrix}. \quad (6)$$

Pascal's equation allows an alternative construction for P_n as,

$$\begin{aligned} p_n^{1,1} &:= 1, \;\; p_n^{2,1} := 1, \;\; p_n^{2,2} := -1, \\ p_n^{i+1,j} &:= p_n^{i,j} + p_n^{i,j-1} \;\; \text{for} \;\; i = 3, 4, \ldots, n, \\ p_n^{i,j} &:= 0, \;\; \text{for} \;\; i = j+1, j+2, \ldots, n. \end{aligned} \quad (7)$$

Indeed, this recursion allows an induction argument to readily confirm the involutary property of P_n.

III. Multinomial coefficients

To lead into the derivations of our main results for this section, consider the polynomials $(s-1)^i$ for $i = 0, 1, 2, \ldots, n$ for some nonnegative integer n and scalar s. Now using the definition of the *Pascal matrix*, it is easily verified that

$$\begin{bmatrix} (s-1)^0 s^n \\ (s-1)^1 s^{n-1} \\ \cdot \\ \cdot \\ (s-1)^n s^0 \end{bmatrix} = P_{n+1} \begin{bmatrix} s^n \\ s^{n-1} \\ \cdot \\ \cdot \\ s^0 \end{bmatrix}. \quad (8)$$

A. A generalized Pascal matrix

Now consider the multinomial (3) for all possible polar binary sequences $\{a_1, a_2, \cdots, a_n\}$. Clearly, the expansion is invariant of the ordering of the a_i, so that there are only $n+1$ selections, namely where there are $k = 0, 1, 2, \cdots, n$ values of $a_i = 1$, with correspondingly $n - k = 0, 1, \cdots, n$ values of $a_i = -1$. Indeed the terms involving sums of products of the a_i in (3) are given, for each $k = 0, 1, 2, \cdots, n$, as the coefficients of the expansion $(s-1)^k(s+1)^{n-k}$. A useful generalization of (8) is then readily established as

$$\begin{bmatrix} (s-1)^0(s+1)^n \\ (s-1)^1(s+1)^{n-1} \\ \cdot \\ \cdot \\ (s-1)^n(s+1)^0 \end{bmatrix} = R_{n+1} \begin{bmatrix} s^n \\ s^{n-1} \\ \cdot \\ \cdot \\ s^0 \end{bmatrix}, \quad (9)$$

for some readily calculated $(n+1) \times (n+1)$ matrix $R_{n+1} := (r^{i,j})$ consisting of elements $r^{i,j}$, and termed here a *generalized Pascal matrix*. In particular, the i^{th} row of R_{n+1} consists of the sums of products of the a_i in (3), for k values of $a_i = 1$, with correspondingly $n-k$ values of $a_i = -1$, and are the coefficients of the polynomial $(s-1)^{k+1}(s+1)^{n-k+1}$.

For reference, the cases for $n = 1, 2, 3, 4$ are spelt out as,

$$R_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (10)$$

$$R_3 = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 0 & -1 \\ 1 & -2 & 1 \end{bmatrix} \quad (11)$$

$$R_4 = \begin{bmatrix} 1 & 3 & 3 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -3 & 3 & -1 \end{bmatrix} \quad (12)$$

$$R_5 = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 1 & 2 & 0 & -2 & -1 \\ 1 & 0 & -2 & 0 & 1 \\ 1 & -2 & 0 & 2 & -1 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix} \quad (13)$$

A recursive relationship between the elements of R_{k+1}, and that of R_k, being a generalization of Pascal's equations, are given for $k = 3, 4, \cdots, n$, initialized by (10),as

$$\begin{array}{rcll} r_{k+1}^{i,1} & := & 1, & \text{for } i = 1, 2, \cdots, k+1 \\ r_{k+1}^{i,j} & := & r_k^{i,j} + r_k^{i,j-1}, & \text{for } j = 2, 3, \cdots, k+1 \\ r_{k+1}^{k+1,j} & := & r_k^{k,j} - r_k^{k,j-1}, & \text{for } j = 2, 3, \cdots, k+1 \end{array} \tag{14}$$

This result is proved in a straightforward manner by induction, and is not spelt out here.

B. Coefficients via the generalized Pascal matrix

As already noted, the multinomial (3), for each possible $a_1, a_2, \cdots, a_n$ selection, is invariant of the ordering of the a_i, and there are then but $n+1$ possible multinomials. These can then be organized as,

$$s_* := \begin{bmatrix} sign_*(n) \\ sign_*(n-2) \\ \cdot \\ \cdot \\ sign_*(-n) \end{bmatrix} = R_{n+1} \begin{bmatrix} \rho_0 \\ \rho_1 \\ \cdot \\ \cdot \\ \rho_n \end{bmatrix} = R_{n+1}\rho. \tag{15}$$

This relationship means that the desired coefficients are the solutions of a linear equation as emphasized in the lemma.

Lemma III.1: The multinomial representation of the $sign_*$ function of (3) has coefficients ρ satisfying the linear equations (15), restated as,

$$R_{n+1}\rho = s_*, \tag{16}$$

where R_n, the generalized Pascal matrix, is defined recursively in (10), and(14).

C. Inverse and decomposition of the generalized Pascal matrix

The nature of the inverse of R_{n+1} now assumes importance. We next develop our second main result, namely that R_n has a factorization in terms of the Pascal matrix P_n, and inherits the involutary property to within a scaling. In particular, we claim,

Lemma III.2: The generalized Pascal matrix R_n, as defined recursively in (10), and(14), has the scaled involutary property

$$R_n^2 = 2^{n-1} I_n, \;\; R_n^{-1} = 2^{1-n} R_n. \tag{17}$$

Proof: This result follows by induction arguments. We work with matrices in lower triangular form. First define F_n as the matrix P_n flipped both left to right and top to bottom. In obvious notation, we write,

$$F_n := \text{flip}(P_n), \;\; \text{or} \;\; f_n^{i,j} = p_n^{n-i,n-j}. \tag{18}$$

Also, define diagonal matrices, in obvious notation, as

$$D_n := diag\{2^0, 2^1, 2^2, \cdots, 2^{n-1}\}, \;\; S_n := diag(P_n). \tag{19}$$

To proceed with the lemma proof, a decomposition lemma is now stated and proved,

Lemma III.3: The generalized Pascal matrix R_n, as defined recursively in (10) and(14), has the decomposition in terms of triangular and diagonal matrices as

$$R_n = 2^{n-1} S_n F_n D_n^{-1} P_n = P_n D_n F_n S_n. \tag{20}$$

Proof: This follows by induction, which is relatively straightforward because only upper or lower triangular matrices are involved. Our approach is guided by keeping in mind the connection of the matrix elements with polynomial coefficients. Thus an equivalent result to (20) is to post-multiply by the vector $[s^{n-1} s^{n-2} \ldots s^0]'$ and apply both (8) and (9) so that,

$$\begin{bmatrix} (s-1)^0(s+1)^{n-1} \\ (s-1)^1(s+1)^{n-2} \\ \cdot \\ \cdot \\ (s-1)^{n-1}(s+1)^0 \end{bmatrix} = P_n \begin{bmatrix} 2^0(s+1)^n \\ 2^1(s+1)^{n-1} \\ \cdot \\ \cdot \\ 2^{n-1}(s+1)^0 \end{bmatrix},$$

$$= F_n \begin{bmatrix} (2s)^{n-1}(s+1)^0 \\ (2s)^{n-2}(s+1)^1 \\ \cdot \\ \cdot \\ (2s)^0(s-1)^{n-1} \end{bmatrix}.$$

These equations are now in a form that they can be verified by straightforward induction arguments. The pattern of the argument becomes clear in passing from $n = 1$ to $n = 2$, and $n = 2$ to $n = 3$, so that passing from n to $n+1$ is then straightforward. It is necessary to exploit the Pascal equations which are inherent in the Pascal matrix P_n construction, and suitably adjusted for the 'flipped' version F_n. Further details are omitted.
□

The proof of (17) follows from (20) by substitution and noting in turn that S_n, P_n, F_n are each readily verified as involutary. Thus,

$$\begin{array}{rcl} (R_n)(R_n) & = & (P_n D_n F_n S_n)(2^{n-1} S_n F_n D_n^{-1} P_n), \\ & = & 2^{n-1} P_n D_n F_n F_n D_n^{-1} P_n, \\ & = & 2^{n-1} P_n D_n D_n^{-1} P_n, \\ & = & 2^{n-1} P_n P_n, \\ & = & 2^{n-1} I_n. \end{array}$$

□

D. Coefficients from columns of the generalized Pascal matrix

The above LemmasIII.1, III.2 together give our main section result stated as a theorem.

Theorem III.1: The multinomial representation of the $sign_*$ function of (3) has coefficients ρ satisfying the linear equations (15), restated as,

$$\rho = 2^{-n} R_{n+1} s_*. \tag{21}$$

where R_n, the generalized Pascal matrix, is defined recursively in (10), and (14), and satisfies (20) and (17).

This result means that matrix inverses are avoided in calculating coefficients. This becomes significant for large n.

This result for $sign_*(sum)$ functions generalizes trivially to any nonlinear function $f(a_1, a_2, \ldots, a_n)$ which is invariant of the ordering of the a_i. The s_* vector is then replaced by a vector with j^{th} element $f(-1, -1, \ldots, 1, 1, 1, \ldots, 1)$, where there are j elements of the data set being -1, and $n - j$ unity elements.

For completeness, we tabulate the coefficients for low n. Specific relationships between the coefficients are clear from the tables and can be proved by induction arguments, as follows.

In Table I, for the *sign* operation,

$$\begin{array}{rcll} \rho_i^{(n)} & = & 0, & \text{for } i = 0, 1, 3, \ldots \text{and all } n, \\ \rho_i^{(n)} & = & \rho_i^{(n-1)}, & \text{for all } i \text{ and } n = 3, 5, \ldots, \\ sign(\rho_i^{(n)}) & = & -1, & \text{for all } n \text{ and } i = 3, 7, 11, \ldots, \\ sign(\rho_i^{(n)}) & = & 1, & \text{for all } n \text{ and } i = 1, 5, 9 \ldots, \end{array} \tag{22}$$

	n=2	n=3	n=4	n=5	n=6	n=7	n=8
ρ_0	0	0	0	0	0	0	0
ρ_1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{5}{16}$	$\frac{5}{16}$	$\frac{35}{128}$
ρ_2	0	0	0	0	0	0	0
ρ_3	0	$-\frac{1}{2}$	$-\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{5}{80}$	$-\frac{5}{80}$	$-\frac{5}{128}$
ρ_4	0	0	0	0	0	0	0
ρ_5	0	0	0	$\frac{3}{8}$	$\frac{5}{80}$	$\frac{5}{80}$	$\frac{3}{128}$
ρ_6	0	0	0	0	0	0	0
ρ_7	0	0	0	0	0	$-\frac{5}{16}$	$-\frac{5}{128}$

TABLE I

TABLE FOR $sign$ FUNCTION MULTINOMIAL COEFFICIENTS.

	n=2	n=3	n=4	n=5	n=6	n=7	n=8
ρ_0	$\frac{1}{2}$	0	$\frac{3}{8}$	0	$\frac{5}{16}$	0	$\frac{35}{128}$
ρ_1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{5}{16}$	$\frac{5}{16}$	$\frac{35}{128}$
ρ_2	$-\frac{1}{2}$	0	$-\frac{1}{8}$	0	$-\frac{5}{80}$	0	$-\frac{5}{128}$
ρ_3	0	$-\frac{1}{2}$	$-\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{5}{80}$	$-\frac{5}{80}$	$-\frac{5}{128}$
ρ_4	0	0	$\frac{3}{8}$	0	$\frac{5}{80}$	0	$\frac{3}{128}$
ρ_5	0	0	0	$\frac{3}{8}$	$\frac{5}{80}$	$\frac{5}{80}$	$\frac{3}{128}$
ρ_6	0	0	0	0	$-\frac{5}{16}$	0	$-\frac{5}{128}$
ρ_7	0	0	0	0	0	$-\frac{5}{16}$	$-\frac{5}{128}$
ρ_8	0	0	0	0	0	0	$\frac{35}{128}$

TABLE II

TABLE FOR $sign_+$ FUNCTION MULTINOMIAL COEFFICIENTS.

and in Table II, for the $sign_+$ operation,

$$\begin{aligned} \rho_i^{(n)} &= 0, \quad \text{for } i = 1,3,\ldots \text{and } n = 3,5,\ldots, \\ \rho_i^{(n)} &= \rho_i^{(n-1)}, \text{ for all } i \text{ and } n = 3,5,\ldots, \\ \rho_i^{(n)} &= \rho_{i-1}^{(n)}, \text{ for } i = 1,3,\ldots \text{and } n = 3,5,\ldots, \\ sign(\rho_i^{(n)}) &= +1, \quad \text{for all } n \text{ and } i = 0,1,4,5,8,9\ldots, \\ sign(\rho_i^{(n)}) &= 1, \quad \text{for all } n \text{ and } i = 2,3,6,7,10,11,\ldots, \end{aligned} \tag{23}$$

For the case of odd n, there is symmetry in the coefficients. Indeed for this case the coefficients for $sign$ and $sign_+$ are identical (since then $sign_+ \equiv sign$).

We see that Table II can be constructed using these various properties and the entries in Table I. Moreover, all coefficients can be constructed from the subset of Table I, namely the ρ_i^n for i, n odd, $i < n/2$.

It is readily seen that for $n > 2$, and either coefficient selection, in obvious notation, then $\sum_{i=1}^{n+1} P_{n+1}(n,i)\rho_{i-1}^{(n)} = -1$, and $\sum_{i=1}^{n+1} P_{n+1}(n-1,i)\rho_{i-1}^{(n)} = 0$. There are other products of the rows of P_n and ρ vectors which are also 0 or 1 not spelt out.

The generalized Pascal Matrix is the key to the coefficients. It is worth pointing out that although this matrix is not orthogonal, yet induction arguments show that all odd rows are orthogonal to all even rows, so that $R_n R_n'$ has zero i, j entries where i is even and j is odd.

IV. Majority logic coding and decoding

In this section, we first recall known majority logic coding and decoding algorithms, for which we now have complete multinomial representations. Building on these results, new majority logic coding algorithms and useful code properties are presented in the following two sections.

A. Majority logic coding

In communication systems based on majority logic coding and decoding, the multinomial expansion (3) represents the baseband transmitted signal. The a_i of (3) are the product of polar binary message data d_i, and associated known time functions $X_i(t) | t \in [0,\ T)$ defined over a time interval $[0,\ T)$ and derived from ℓ-vector binary codewords, denoted X_i.

Consider a data row vector $d := [d_1\ \ d_2 \ldots d_n]$ and codewords as row vectors of length ℓ, as X_i. Denote a code matrix X as an $n \times \ell$ matrix with rows X_i. The time functions representing the encoded data for baseband transmission, are constant within (equal) time slots called chips. They are defined over a time interval T of $\ell \geq n$ chips. In this case the coded transmitted signals are, in continuous time $s_*(\cdot)$, or in discrete time $s_* := [s_1\ \ s_2 \ldots s_\ell]$,

$$\begin{aligned} s_*(t): &= sign_*\left(\sum_{i=1}^{n} d_i X_i(t)\right), \\ s_*: &= sign_*(dX). \end{aligned} \tag{24}$$

Notice that for n even, then application of the $sign$ function leads to ternary transmitted signals, but for n odd they are polar binary, as for the $sign\pm$ for all n. The simplicity of the majority logic encoding operation is a key virtue, which also carries over to the decoding operation.

Consider the multinomial expansion for s and for $s_\pm$ in the discrete time case, namely (3) with the a_i replaced by dX_i. The coefficients are the same for n odd, see also the Tables I and II. In the case n even, and with X orthogonal in that $XX' = I_n$, the coefficient squared of each component represents the energy contribution from each component. For the $sign$ case, s_i can be zero and still carry information, and there is no transmission of the product $d_1 d_2 \cdots d_n$.

B. Majority logic decoding

At the receiver, the received signal $r(t)$ is the transmitted signal plus added noise $w(t)$, as $r(t) = s_*(t) + w(t)$. For decoding to estimate d_i, one can work with an optimal *maximum likelihood decoder*, but the complexity grows as 2^n, since there must be a decorrelation for each of the 2^n possible data sets $d_1, d_2, \ldots, d_n$.

There is incentive then to consider a much simpler decoding process termed a *majority logic decoding*, as

$$\widehat{d_i} := sign_+\left(\int_0^T \frac{(r(t) - \rho_0)}{n\rho_1} X_i(t) dt\right). \tag{25}$$

Here, we use $sign_+$, rather than $sign_*$ to ensure that the estimate is bipolar binary.

Experimentally, despite the danger of losing information in the majority logic operations of the coding and decoding process there are codes, termed *majority logic codes*, which achieve error-free decoding in the noise free case, in that $\widehat{d_i} = d_i$ when $r(t) = s_*(t)$. Moreover, in the added noise case, for sufficiently small noise, the $sign_*$ operation preserves error free coding/decoding in that $\widehat{d_i} = d_i$. How can such codes work? How can such codes be constructed and optimized? How can the decoding of such nonlinear codes be simplified?

C. Error-free decoding in the noise-free case

Different majority logic codes achieve differing robustness to noise, but in order to be considered at all, it is a usual requirement that they have no deterministic errors. That is, denoting a matrix D of all 2^n possible bipolar data D_i in its rows, then

$$D = sign_\pm\left(sign_*(DX)X'\right). \tag{26}$$

This equations (26) can be used to search for suitable codes, in that random or other selections of X can be tested for suitability.

A simply proved, but very useful result is the following,

Lemma IV.1: Consider a code matrix X which achieves error free majority logic coding and decoding in the noise-free case, in that (26) holds. Then any signed row or column permutation of this matrix preserves this property of no deterministic coding errors. That is, denoting signed permutation matrices of dimension n as Π_n,

$$D = sign_{\pm}\left(sign_*(D[\Pi_n X \Pi_\ell])[\Pi_n X \Pi_\ell]'\right). \tag{27}$$

Moreover, any one $n \times \ell$ code matrix X can be a basis for generating up to $2^{(\ell+n)}\ell!n!$ code matrices with this property.

Proof: Any signed re-ordering of the rows of X is equivalent to a signed re-ordering of the data, for which no deterministic errors are introduced, by assumption. That is, since $[D\Pi_n']$ is a valid data set, and $\Pi_n'\Pi_n = I_n$, then from (26),

$$[D\Pi_n'] = sign_{\pm}\left(sign_*([D\Pi_n'][\Pi_n X])[\Pi_n X]'\right). \tag{28}$$

Also, signed re-orderings on the columns of X for the coding process are matched in the decoding process, so there is no total effect. That is,

$$D = sign_*\left(sign_{\pm}(D[X\Pi_\ell])[X\Pi_\ell]'\right). \tag{29}$$

□

D. Multinomial representation of decoding

To further understand majority logic decoding, here we build on the work of [1]. We consider first the noise-free case when $r(\cdot) = s_*(\cdot)$.

In $\widehat{d_i}$ of (25), let us substitute the multinomial representation (3) for the $sign_*$ function $s_*(t)$ of (24). We consider one term of the multinomial at a time, and indeed break each term down further and consider one summation component at a time. That is, we consider each possible integral of partial products one at a time for $i = 1,\ \ 2, \dots n$ as,

$$\begin{aligned}\widehat{d_i} = \quad & sign_*[d_1 \textstyle\int_0^T X_1(t)\frac{X_i(t)}{n}dt + d_2 \int_0^T X_2(t)\frac{X_i(t)}{n}dt + \dots \\ & + \tfrac{\rho_2}{\rho_1} d_1 d_2 \textstyle\int_0^T (X_1(t)X_2(t))\frac{X_i(t)}{n}dt + \dots \\ & + \dots \\ & + \tfrac{\rho_n}{\rho_1} d_1 d_2 \dots d_n \textstyle\int_0^T (X_1(t)X_2(t)\dots X_n)\frac{X_i(t)}{n}dt].\end{aligned} \tag{30}$$

Let us denote a data vector d augmented with partial products as d^{pp}, a corresponding coefficient vector as ρ^{pp} and a codeword function vector $X(\cdot)$ augmented with corresponding element by element partial products as $X^{pp}(\cdot)$, and define these from,

$$\begin{aligned} d^{pp} \quad &:= \quad [d_1 \ \ d_2 \dots d_n, \ \ (d_1 d_2) \dots \\ & \qquad (d_{n-1}d_n), \dots, (d_1 d_2 \dots, d_n)]', \\ \rho^{pp} \quad &:= \quad [\frac{\rho_1}{\rho_1} \ \ \frac{\rho_1}{\rho_1} \dots \frac{\rho_1}{\rho_1}, \ \ \frac{\rho_2}{\rho_1} \ \ \frac{\rho_2}{\rho_1} \dots \frac{\rho_2}{\rho_1}, \ \ \dots, \frac{\rho_n}{\rho_1}]', \\ X^{pp}(\cdot) \quad &:= \quad [X_1(\cdot)' \ \ X_2(\cdot)' \dots X_n(\cdot)', \ \ (X_1(\cdot).X_2(\cdot))' \dots \\ & \qquad (X_{n-1}(\cdot).X_n(\cdot))', \dots, (X_1(\cdot).X_2(\cdot). \dots . X_n(\cdot))']'. \end{aligned} \tag{31}$$

The number of distinct partial products of j data bits, or code words is ${}^nC_j$, so that the numbers of identical elements in ρ^{pp}, separated by commas in (31), are then given from the binomial coefficients as ${}^nC_1, {}^nC_2, \dots, {}^nC_3$, the sum of which is $2^n - 1$. Thus d^{pp}, ρ^{pp} are $2^n - 1$ row vectors.

Let us denote the rows of $X^{pp}(\cdot)$ as $X_i^{pp}(\cdot)$. When the codewords are represented as an $n \times \ell$ matrix X, then denote an augmentation of this as a $2^n \times \ell$ matrix X^{pp}, with rows X_i^{pp}. Of course, $X = [\underline{1}_n' \ \ 0 \ \ 0 \dots 0]X^{pp}$, where $\underline{1}$ is a vector of n unity elements.

Now the multinomial expansion (30) can be written for $i = 1,\ \ 2, \dots n$ as,

$$\widehat{d_i} = sign_*[\int_0^T \frac{X_i^{pp}(t)X^{pp}(t)'}{n}dt \ \ diag\{\rho^{pp}\}d^{pp}]. \tag{32}$$

This is essentially the same as the discrete time case when

$$\widehat{d_i} = sign_*(\frac{X_i^{pp}(X^{pp})'}{n} \ diag\{\rho^{pp}\}d^{pp}). \tag{33}$$

We for a vector x and matrix X,we denote $diag\{x\}, diag\{X\}$ as a diagonal matrices with i^{th} diagonal elements x_i, X_{ii} ,respectively.

V. Partial product estimation for enhanced decoding

In this section, we first recall known concepts to reduce majority logic decoding errors in the presence of transmission channel noise, using partial product estimation. Building on this work we give new results for noise error reduction based on explicit multinomial expansions, and then propose new majority logic coding algorithms to assist this process.

A. Estimation of data partial products

Partial products $d_i, d_i d_j, d_i d_j d_k, \dots$, can be estimated via a generalization of (25), for $i = 1,\ \ 2, \dots 2^n - 1$ as,

$$\widehat{d_i^{pp}} := sign_*\left(\int_0^T \frac{(r(t) - \rho_0)}{n\rho_1}(\rho_i^{pp})^\dagger X_i^{pp}(t)dt\right). \tag{34}$$

Here $\alpha^\dagger$ denotes α^{-1} when the inverse exists and zero otherwise. Clearly, the partial products with zero coefficients ρ_i^{pp} in the multinomial expansion can not be estimated: Our equations set these products and their estimates to zero.

The partial products of d_i, are in fact parity bits, so that the vector d^{pp} can be viewed as message bits together with parity bits. The multinomial expansion of d^{pp} in the noise free case, in obvious notation, is

$$\begin{aligned} Y(X) \quad &:= \quad \int_0^T diag\{(\rho^{pp})^\dagger\}X^{pp}(t)X^{pp}(t)'diag\{\rho^{pp}\}dt, \\ & \sim \quad diag\{(\rho^{pp})^\dagger\}X^{pp}(X^{pp})'diag\{\rho^{pp}\}, \\ \widehat{d^{pp}} \quad &= \quad sign_*\left(Y(X)d^{pp}\right). \end{aligned} \tag{35}$$

It is clear that the partial product estimates may not be free of errors, even in the noise free case, unless Y is suitably diagonally dominant; more on this in a later section. It is also clear that for the *sign* operation, the multinomial does not contain partial products of an even number of data, so that these can not be estimated with conventional coding. We propose a novel coding to facilitate the estimation of partial products as follows.

B. Majority logic coding of data and partial products

Here we propose a majority logic coding scheme which comprises a first stage of linear coding with its own spreading factor, followed by a second compatible stage of majority logic coding with its own spreading factor, and a single stage majority logic decoding to recover the data. That is, we consider the majority logic coding (24) and decoding (25), but with the data d_i preprocessed in a linear coding stage.

In particular, here the linear preprocessing stage is simply the augmentation of the data with partial products. The result is denoted d_i^p, which can spread to the full complement of partial products d_i^{pp}. Also, codewords X_i are replaced by codewords augmented with corresponding partial products X_i^p, up to the full complement X_i^{pp} as,

$$\begin{aligned} s_*(t): &= sign_*\left(\sum_{i=1}^{n} d_i^p X_i^p(t)\right), \\ s_*: &= sign_*(d^{pp}X^p), \end{aligned} \tag{36}$$

$$\widehat{d_i^p} := sign_+\left(\int_0^T \frac{(r(t)-\rho_0)}{n\rho_1} X_i^p(t)dt\right). \tag{37}$$

These will give error free decoding in a noise free environment if, denoting the matrix of all possible data vectors D augmented with the partial products as D^p,

$$D^p = sign_\pm\left(sign_*(D^pX^p)(X^p)'\right). \tag{38}$$

We stress that the nonlinear stage only codes for 2^n codewords, even though its input is n data bits plus parity bits.

In the noisy channel case, of course a maximum likelihood decoding of the data can be simply implemented, but this tends to be prohibitive as the data size and spreading factors increase to practical values. Consequently, a two stage decoding can be applied as follows:

First Stage: Apply the standard decoding to recover estimates of the data augmented with partial products as $\hat{d}^p$. These can be hard decision estimates using the usual linear operation followed by the $sign_+$ operation, or preferably soft decision estimates from just the linear operation without taking the $sign_+$ operation.

Second Stage: Use standard decoding to recover the data estimates from $\hat{d}^p$. For example, first estimate each data bit d_i from each of the entries in the augmented vector $\hat{d}^p$ which are dependent on d_i, using whatever knowledge is available on the other relevant bits. Then add these estimates and take the $sign_+$ operation which is a majority logic operation. Thus, if there is a parity bit d_1d_2, then estimating d_1 from this one can use in obvious notion $\widehat{d_1\hat{d}_2}$, where $\hat{d}_2$ is the currently available, preferably soft, estimate of d_2.

We note an illustrative example of where this extended majority logic approach is immediately useful. In the case of majority logic coding for five data spreading to eight chips, there is no code which gives error free decoding in the noise free case. However, by augmenting with one or two parity bits, error free decoding of the data and parity bits can be achieved. In contrast, for the case of three data bits spreading to eight chips , our simulation studies indicate no advantage in augmenting with parity bits in a preprocessing linear coding stage. However, including the linear preprocessing allows a "more satisfactory" analysis of the nonlinear coding, since the nonlinear coding applies to less spreading.

C. Correcting data estimation errors using the product parity bit estimate

An approach proposed in [1] to avoid *maximum likelihood decoding* in continuous time, when *estimates of the data and the product parity bit is available* , is now recalled. It aims to correct one bit estimation error in a data vector estimate, assuming that there is only one such error.

Prior to taking the $sign_*$ operation to achieve d^p estimates, select the particular value which has least magnitude, but only in the case that this is a data bit estimate. Assume the $sign_+$ of this value is a bit error, but that all other bit estimates and partial product estimates are correct. Then re-estimate this bit from the remaining $\hat{d}_j^p$, and the partial product estimate. This approach works well in low noise. As the noise level increases, this approach introduces additional errors into the system.

VI. A diagonal dominance property for majority logic codes

In this section, we seek to characterize codes for majority logic coding, which are free of deterministic errors. Of course, random or systematic search or numerical test procedures can be used, to ensure that (26), or (38) is satisfied. This requires of order $2^{n\ell}$ calculations for $n \times \ell$ code matrices X. Also, we develop theory based on *diagonal dominance* properties of $Y(X)$ of (35): This matrix arises from the multinomial representation of the majority decoding process, and is a simple function of the code matrix X, and multinomial coefficients ρ. First, stronger properties than diagonal dominance, namely orthogonality properties are considered.

A. Codeword orthogonality properties

Many of the insights used in the design of majority logic coding schemes come from thinking in terms of codeword orthogonality. Such conditions can only apply for the case of n even, but they can apply approximately for n odd. We consider first codeword orthogonality, then extended codeword orthogonality and later 'relaxed orthogonality'.

Zero mean property: This is essentially an orthogonality of the codeword matrix X to a constant as,

$$X\underline{1}_\ell = 0. \tag{39}$$

where $\underline{1}_\ell$ denotes an ℓ column vector of unity elements. Consider the case when $sign_\pm$ is used with n even, when $\rho_0 \neq 0$. Now if the zero-mean condition (39) holds, then the multinomial expansion for $\widehat{d_i}$ is invariant of ρ_0, so we can take $\rho_0 = 0$ as in the case n odd.

This zero-mean property on X excludes generating any code word from the first row of the Hadamard matrix which is all $+1$.

Relaxed codeword orthogonality property: Of course, if the codewords are orthogonal to each other, then $XX' = nI_n$, or if the codewords are extended by parity bits of partial products, then the relevant orthogonality condition is that $X^{pp}(X^{pp})' = (2^n-1)I_{(2^n-1)}$. For our purposes we relax this strict definition as the matrix $Y(X) := (Y_{i,j}(X))$ being diagonal, that is

$$Y_{i,j}(X) = 0, \text{ when } i \neq j. \tag{40}$$

A further relaxation of the orthogonality of $Y(X)$, can be expressed in terms of an indicator matrix $\mathcal{I}$, which is unity in selected diagonal elements and zero otherwise, as

$$(\mathcal{I}_{(2^n-1)}Y(X))_{i,j} = 0, \text{ when } i \neq j. \tag{41}$$

Of course, an important special case is when $\mathcal{I}_{(2^n-1)} = diag\{\underline{1}_n' \ 0 \ 0 \ldots 0\}$, which leads to the condition that the code matrix X is orthogonal. If $\mathcal{I} = I$, then (40) is recovered.

B. 'Ideal' code from Hadamard matrices

An 'ideal' code having n codewords can be envisaged, which would have the set of all $2^n - 1$ possible products $X_i, X_iX_j, X_iX_jX_k, \ldots$ *distinct and orthogonal*, that is (40) holds. There are at least $\ell = 2^n - 1$ chips for such a code. The Hadamard matrix is an orthogonal matrix, known to have

rows which are Walsh functions with the property that partial products of rows are also Walsh functions. Thus for any $2^n \times 2^n$ Hadamard matrix any subset of less than or equal to n rows form a tentative codeword matrix and whether or not this is extended with its partial products it can be readily verified that the desired orthogonality property holds.

As an example, consider $n = 3, 2^n - 1 = 7$, and now focus on the 8×8 Hadamard matrix, denoted $H(8)$. Take $n = 3$ code words of length $\ell = 8$, as the rows $2, 5, 8$ of this matrix, to form the code matrix

$$X = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}. \quad (42)$$

Then the partial products of pairs of these codewords constitute the rows $4, 6, 7$ of $H(8)$, and the triple product is row 3. Clearly, altogether the rows constitute all but the first row of $H(8)$, and so are orthogonal with $X^{pp}(X^{pp})' = 8I_7$.

Taking another matrix row as an additional codeword, this row and its attendant partial products, then these are not all distinct from the earlier chosen code words and their partial products. Thus, such sets of four, or more, codewords extracted from the 8×8 Hadamard matrix, can not constitute an 'ideal' code .

In the case of an 'ideal' code, with (40) holding, then all the summands in the multinomial expansion for $\widehat{d_i}$ involving any $d_j|_{j \neq i}$ would be zero, and there is then guaranteed recovery of d_i in the noise free case. (Such recovery is not excluded with more codewords than in the 'ideal' case.)

Further, for an 'ideal' code, all the partial products $d_i, d_i d_j, d_i d_j d_k, \ldots$ can be estimated via (34). The orthogonality condition (40) is satisfied, $Y(X)$ is a diagonal matrix and trivially the following diagonal dominance property holds

$$diag\{abs(Y(X))\underline{1}_\ell\} < 2diag\{Y(X)\}.$$

Here $abs(Y(X))$ denotes a matrix $Y(X)$ with elements replaced by absolute values. Consequently, in the noise-free case, there are no deterministic errors in the decoding of all partial products. (Such full partial product recovery appears excluded in the case of more codewords than in the 'ideal' case, although recovery of the data and perhaps one partial product can be achieved in some cases.)

In the case when the received signal has additive noise, the parity bit estimates allow an improved estimation of the message bits, and can be exploited to enhance estimation.

Our numerical studies lead us to confidently conjecture a new result for which we do not have a theoretical proof.

Lemma VI.1: With the majority logic coding and decoding of data with partial products as in (36),(37), based on Hadamard codeword matrices satisfying (40), then there are no deterministic errors.

Can the orthogonality condition be relaxed?

C. Diagonal dominance property

A useful approximation to the orthogonality property (41) for our purposes is a specific 'diagonal dominance' of $Y(X)$ of (35) as,

$$\mathcal{I}_{(2^n-1)} diag\{abs(Y(X))\underline{1}_\ell\} < 2\mathcal{I}_{(2^n-1)} diag\{Y(X)\}. \quad (43)$$

where the inequality is assumed to include the case $0 < 0$. Thus, the following lemma is readily established.

Lemma VI.2: Consider a code matrix X and $Y(X)$ given from (35). Then the diagonal dominance condition (43), for some indicator matrix $\mathcal{I}_{(2^n-1)}$, ensures that there are no deterministic errors in estimating $d^p := d^{pp}\mathcal{I}_{(2^n-1)}$ via (35), in that (38) holds. Moreover, the diagonal dominance condition is implied by the stronger orthogonality condition (41).

The sufficiency of the data-independent diagonal dominance condition (43) is clear. This condition is by no means necessary, since the data and partial products are not independent. However, it is not straight forward to achieve a data independent necessary condition. Checking for deterministic errors for all possible data is perhaps the only option, but this grows exponentially with the data length. The usefulness or otherwise of the sufficiency condition (43) is illustrated for codes related to the Hadamard codes as follows.

D. 'Near ideal' codewords from Rademacher matrices

A Rademacher matrix is simply the Hadamard matrix with the first row and column deleted, and can be used to generate n majority logic codewords of length $\ell = 2^n - 1$. Consider the code matrix X of (42), but with the first column deleted, denoted $\overline{X}$, and augmented with partial products as $\overline{X}^{pp}$.

Now $\overline{X}^{pp}(\overline{X}^{pp})'$ is not orthogonal, but its construction ensures the property $\overline{X}^{pp}(\overline{X}^{pp})' = (n)I_{(n-1)} - \underline{1}_{(n-1)}\underline{1}'_{(n-1)}$. However, our numerical studies for Rademacher matrices up to $n \leq 8$ show that $Y(\overline{X})$ is diagonally dominant as defined in (43), for n odd. For n even, a relaxed dominance condition can be satisfied in that (43) holds but with equality for some i.

Our numerical studies for $n \leq 8$ show that certainly for some Rademacher matrices, although the diagonal dominance condition fails, there are invariably no deterministic errors in the noise free case, in that (38) holds. This illustrates the fact that the diagonal dominance condition is only a sufficient condition, not a necessary one. Clearly, this example also illustrates that the gap between necessity and sufficiency will reduce if the even numbered coefficients ρ_i are zero, as when n is odd.

Our numerical studies lead us to confidently conjecture a new result,but so far without theoretical justification save for the case of codeword matrices satisfying (43),

Lemma VI.3: With the majority logic coding and decoding of data with partial products as in (36),(37), based on Rademacher codeword matrices, then there are no deterministic errors.

E. 'Near ideal' codewords from PN sequences

Pseudo-random noise (PN) sequences of bipolar binary bits of length $2^n - 1$ and satisfying the so-called correlation properties, see [8], can generate codewords.

Consider a sequence of length $2^n - 1$. Now form a matrix, where each row is a cyclically shifted version of the previous row. These have identical diagonal dominance properties as those generated from the Rademacher matrices. Once the matrices are generated, n codewords can be selected from the rows, which form a basis, in that the remaining rows can be formed by partial products of the codewords to form a matrix X^{pp} such that the diagonal elements of $X^{pp}(X^{pp})'$ are $2N - 1$, but the off diagonal elements are either $+1$ or -1, which is different from the Rademacher case. Then the row sums of the $Y(X)$ matrix are the same as for the Rademacher codes. Our numerical studies confirm that the diagonal dominance condition is not necessary for error free deterministic decoding in cases for $n \leq 8$, although we are not aware of any theoretical proof. Thus,we confidently conjecture a new result, but so far without theoretical justification save for the case of codeword matrices satisfying (43),

Lemma VI.4: With the majority logic coding and decoding of data with partial products as in (36),(37), based on *pseudo-*

random codeword matrices, then there are no deterministic errors.

F. 'Near ideal' codewords from the Wing code

We observe here that particular Wing code sets (matrices) of [9] can be generated from a Rademacher matrix of dimension $2^n - 1$ with any one column inverted. Consider, for example, such a selection of three codewords. Take for example rows $3, 5, 6$, of a Rademacher 7×7 matrix, with the last two columns interchanged. This achieves a Wing code, but taken together with its extension of partial products it does not constitute a Wing matrix, but rather generates a Rademacher matrix with a column interchange. This being the case, a diagonal dominance condition is satisfied for this set of three codewords, so that these will not generate deterministic errors in a majority logic coding.

G. Low processing gain code words

The spreading factor of any 'ideal' or 'near ideal' code, such as those based on the Hadamard matrix, Rademacher and PN sequences as above, would be $\frac{2^n}{n}$,or $\frac{2^n-1}{n}$. This is considered too high unless n is a 'small' integer, such as $2, 3, 4$.

With $\frac{\ell}{n} << 2^n$, instead of all code words and their sums of partial products being orthogonal to one another, some pairs of this set will not be orthogonal.

It appears preferable for robust decoding to have the diagonal dominance condition (43) holding, and indeed to maximize in some sense diagonally dominance or the inequalities in (43).

One approach is to seek to maximize the number of pairs of partial codeword products that are orthogonal, or equivalently, to achieve maximum sparcity in $Y(X)$ of (35).

A second approach is to minimize the maximum relevant row sum of $abs(Y(X))$.

Special cases are of interest and for some of these we note additional properties of the coefficients ρ.

H. Coefficients ρ^{pp}

Using the formula (15), it is verified that

$$sign(\frac{\rho_n}{\rho_1}) = \begin{cases} 0 \text{ for } n \text{ even}, \\ -(1)^{\frac{n}{2}}, \text{ for } n \text{ odd}. \end{cases} \quad (44)$$

Also, $S_{rem} := \sum_{j=2}^{n-1} \frac{|\rho_j|}{|\rho_1|} < 1$ is a monotonically decreasing value as n increases, for either n even or odd, at least for $n > 4$. An induction argument can be developed for these properties. For reference, we list some S_{rem} values,

S_{rem}	$sign$	$sign_+$	S_{rem}	$sign$	$sign_+$
n=2	0	0	n=3	0	0
n=4	0.3333	0.6667	n=5	0.3333	0.3333
n=6	0.4000	0.8000	n=7	0.4000	0.4000
n=8	0.3714	0.7429	n=9	0.3714	0.3714
n=10	0.3175	0.6439	n=11	0.3175	0.3175
n=12	0.2641	0.5281	n=13	0.2641	0.2641
n=14	0.2191	0.4382	n=15	0.2191	0.2191
n=16	0.1835	0.3671	n=17	0.1835	0.1835
n=18	0.1560	0.3120	n=19	0.1560	0.1560

I. Orthogonal codeword augmented with codeword products

Let us consider the case when at least the codeword matrix augmented with the product of all codewords is orthogonal. Then the first n summands in the multinomial expansion (30) simplify as d_i, and the last term is zero. What influence do the remaining terms have?

Now consider in addition, that all but one of ${}^nC_j$ entries in a row X_i with identical coefficient ρ_j is zero. The explicit formula for the coefficients (15), allows us to verify, at least numerically, that in the multinomial expansion for $\widehat{d}_i$ the summation of terms involving the ρ_j coefficients for $j \neq 1, n$, denoted S_{rem} earlier, are in magnitude less than 1, so there is diagonal dominance with (43) holding, and $\widehat{d}_i = d_i$. That is, under our orthogonality assumptions, the sum of the magnitudes of remaining terms satisfies is $S_{rem} := \sum_{j\neq 1,n}^{n} \frac{|\rho_j|}{|\rho_1|} < 1$.

Consider a useful example of an 8×8 Hadamard matrix with the first row deleted. Now for n codewords defined as certain $n = \{4, 6, 7\}$ row selections of this matrix, the orthogonality property (43) holds, so that there are no deterministic errors for the data estimates in these cases. Moreover, there are no deterministic errors in estimating the product of all codewords using the $sign_\pm$ function, or the $sign$ function for $n = 7$. For the case of $n = 5$ codewords, the orthogonality condition fails, and there is always one data bit that can not be estimated without deterministic errors, and the product of codewords can not be estimated. (The cases of $n = \{1, 2, 3\}$ code words would come under the heading 'ideal' codewords, as earlier.)

Another useful example are 3 codewords constructed from the rows of a 4×4 Hadamard matrix with the first row deleted. Again, the dominance condition is satisfied and there are no errors in estimating the data or product of the data in the noise-free case.

J. Other examples of error-free majority logic coding

Of course, the conditions for error-free majority logic decoding explored above can be relaxed, but the orthogonality assumptions are then more detailed. Also, it becomes more difficult to find codes which satisfy such conditions.

VII. Comparative Performance in White Noise

In this section we graph the bit error rate performance for example majority logic coding schemes presented in this paper in the presence of additive white Gaussian noise (AWGN), see Figure 1. We consider majority logic coding schemes with $n = 3$ and $\ell = 8$. The legend 'ML Type A' represents the plot for a conventional majority logic coding scheme of Section IV. Legend 'ML Type B' represents the BER performance plot for the majority logic scheme proposed in Sub-section V.B, using an additional 3 linear parity bits. The legend 'ML Type C' represents the performance of the majority logic coding scheme introduced in [1], where a parity bit is estimated using the multinomial representation. The legend 'Max Likelihood' represents the computationally intensive optimal decoding.

From Figure 1, when the proposed majority logic coding scheme with partial parity is used, we observe an almost identical coding performance to the conventional scheme. Clearly, in this case, the gain achieved by the additional linear partial parities are offset by the loss in nonlinear code strength.

We can observe that the low noise assumption made by the 'ML Type C' scheme of [1] is not helpful in that the performance is not even as good as that of the conventional majority logic scheme. The loss in performance is due to the fact that under conditions of high noise, the probability of error in the parity bit is quite high thus, when used together in the information detection process will result in a higher error rate. On the other hand, under extremely low noise conditions, the additional parity bit will only add to the strength of the detection process, hence giving a better overall bit error rate performance.

These studies illustrate that there is still the need to test any proposed codes before deciding if they will help.

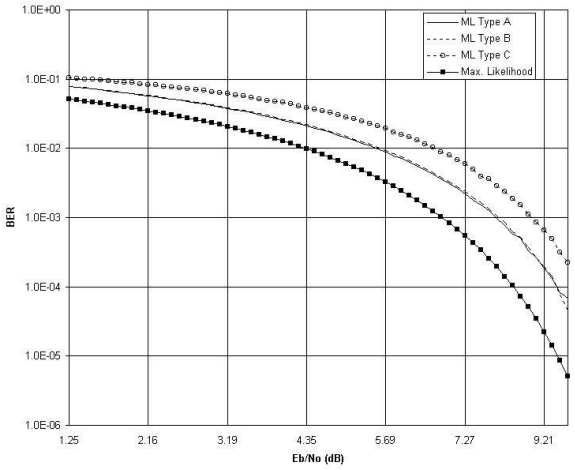

Fig. 1. BER performance of majority logic coding schemes over the AWGN channel.

VIII. Conclusions

Majority logic coding for communication systems has attractive advantages in terms of the simplicity of the decoding. This is achieved at the expense of optimality. The majority logic operations involved are highly nonlinear, so there has been a paucity of theory for developing codes and guaranteeing properties.

A key step in this direction taken in this paper has been the generation of an explicit, formula for the multinomial representation of the various $sign_*$ operations involved in majority logic. The formula is readily calculated in terms of binomial coefficients, appearing in a proposed generalized Pascal matrix. A factorization of this matrix in terms of a lower Cholesky factor of the original Pascal matrix turns out to simplify the proof and derivation of the coefficients. The results are more complete than hitherto given for the case of $sign$, and are new for the $sign_\pm$ case.

The next step has been the proposal of a diagonal dominance condition on functions of any codeword and the multinomial expansion coefficients. This diagonal dominance is proved to be a sufficient condition for guaranteeing error-free coding and decoding in a noise-free environment. Its usefulness has been demonstrated in guaranteeing deterministic error-free coding and decoding for known classes of majority logic codes: In particular codes base on Hadamard matrices, Rademacher matrices, and *pseudo*-random sequences. Previously, only numerically verified versions of such results via exhaustive search have been available for low dimensional cases.

Finally, new majority logic coding algorithms have been proposed which are essentially two stages of codings, the first being a linear stage and the second a compatible majority logic stage. The majority logic stage is based on code matrices extended with partial products to be applied to data extended with partial products. The two stage codes and corresponding majority logic decoding, appear to work as well as or better than single stage codes, and at least the linear stage is more amenable to analysis.

References

[1] V.P. Ipatov, Y.A. Kolomensky, and R.N. Shabalin Reception of Majority-Multiplexed Signals. *Radio Engineering and Electronic Physics*, vol. 20, no. 4, pp.121-124, 1975.

[2] R.C. Titsworth Application of the Boolean for the Design of a Multi-Channel Telemetric System (in Russian). *Zarubezhnaya Radioelektronika*, 8, 1964.

[3] R.C. Tistsworth A Boolean-Function-Multiplexed Telemetry System *IEEE Transactions on Space Electronics and Telemetry*, vol. SET-9, pp42-45, June, 1963.

[4] J.A. Gordon and R. Barrett Correlation-recovered adaptive majority multiplexing *Proceedings of IEE*, vol. 118, no. 3/4, pp.417-422, 1971.

[5] A.K. Mukherjee and D. Mukhopadhyay A method for increasing the number of majority multiplexed channels *Proceedings of IEEE*, vol. 66, no. 9, pp.1096-1097, September, 1978.

[6] T. Maseng Performance Analysis of a Majority Logic Multiplex System *IEEE Transactions on Communications*, vol. COM-28, no. 9, September 1980.

[7] K.T. Tan, R. Liyanapathirana and K. N. Ngan Error probabibities for sequency majority multiplexing in frequency-nonselective, slowly fading channel - Part 1 & 2 *Proceedings of IEEE 5th ISSSTA*, pp.411-419, September, 1998.

[8] Golomb, Solomon W. *Shift Register Sequences*. Holden-Day, Inc. San Francisco, 1967.

[9] P.A. Wing Code Division Multiplexing. *Monitor-Proc.IREE*, 1, pp.25-28, 1976.