Electrical Engineering and Computer Science - Technical Reports

College of Engineering and Computer Science

7-1969

Generalized Threshold Decoding of Convolutional Codes

Luther D. Rudolph
Syracuse University

Recommended Citation

Rudolph, Luther D., "Generalized Threshold Decoding of Convolutional Codes" (1969). *Electrical Engineering and Computer Science - Technical Reports*. 15.
https://surface.syr.edu/eecs_techreports/15

GENERALIZED THRESHOLD DECODING OF CONVOLUTIONAL CODES

LUTHER D. RUDOLPH

JULY, 1969

SUOS
CULTORES
SCIENTIA
CORONAT

SYRACUSE UNIVERSITY : FOUNDED A·D·1870·

SYSTEMS AND INFORMATION SCIENCE
SYRACUSE UNIVERSITY

GENERALIZED THRESHOLD DECODING OF CONVOLUTIONAL CODES

L.D. RUDOLPH

SYSTEMS AND INFORMATION SCIENCE

SYRACUSE UNIVERSITY

JULY, 1969

ABSTRACT

It is shown that any rate $1/b$ systematic convolutional code over GF(p) can be decoded up to its minimum distance with respect to the decoding constraint length by a one-step threshold decoder. It is further shown that this decoding method can be generalized in a natural way to allow "decoding" of a received sequence in its unquantized analog form.

FOOTNOTES

GENERALIZED THRESHOLD DECODING OF CONVOLUTIONAL CODES

I. Introduction

Majority decoding of linear block codes using parity checks that are orthogonal, or can be orthogonalized in steps, was first proposed by Reed [1] in 1954. This idea was further developed by Massey [2] and Mitchell et al. [3] in the early 1960's. Subsequently, Rudolph [4] proposed a one-step majority decoding scheme based on parity checks that are in general not orthogonal. In 1968, Weldon [5] presented improved majority decoding algorithms for block codes associated with finite geometries. Recently, two generalized threshold decoding methods have been devised that are capable of decoding any binary or nonbinary linear block code up to its minimum distance. The first scheme, devised by Gore [6], is a generalization of Massey's L-step orthogonalization, and achieves additional decoding power by relaxing the requirement that parity checks be strictly orthogonal. The second scheme, proposed by Rudolph [7], is a generalization of one-step majority decoding based on non-orthogonal parity checks and achieves its added power by replacing the majority element by a more general threshold element.

Threshold decoding of convolutional (recurrent) codes was introduced by Massey [2] in 1962. He showed that multiple-step threshold decoding is not applicable to convolutional codes (in the sense that any convolutional code that is L-step orthogonalizable is one-step

orthogonalizable). Thus, generalized L-step orthogonalization does not carry over to convolutional codes in a natural way. One purpose of this paper is to show, however, that generalized one-step threshold decoding based on non-orthogonal parity checks does carry over. (Although this technique applies to any convolutional code, only rate $1/b$ codes in systematic form will be considered for ease of presentation.)

The second purpose of the paper is to show that one-step threshold decoding can be generalized in a very natural way to allow "decoding" of a received sequence in its unquantized analog form. This generalized decoding technique, which we call analog threshold decoding, is particularly well suited to the processing of convolutional codes because of the continuous nature of the information flow.

This paper is organized as follows. In Section II, the decoding problem for convolutional codes is formulated in a way suited to the development to follow. In Section III, the Fourier series representation of discrete functions is reviewed and then applied to show the existence of a generalized one-step threshold decoder for convolutional codes. Examples are presented to illustrate an approach to the decoder synthesis problem. In Section IV, the extension to analog threshold decoding is discussed and two methods of implementing analog threshold decoders are suggested. Concluding remarks are contained in Section V.

II. Decoding Problem for Convolutional Codes

In what follows, we will consider only rate 1/b convolutional codes in systematic (canonical) form over $GF(p)$, the field of p elements, p a prime. All operations in this section are performed over $GF(p)$. The notation used is essentially that of Wyner and Ash [8] and Wyner [9].

A rate 1/b systematic convolutional code is the null space of a semi-infinite parity check matrix H of the form shown in Figure 1. All the nonzero entries of H are in the shaded areas. A sequence (semi-infinite row vector) x is a code word if and only if $xH^T = 0$. The code is said to have block length b with $m = b-1$ check symbols per block. If $v = 1$, then H is the parity check matrix of a block code.

A received sequence y is the sum of the transmitted code word x and a semi-infinite error vector e, i.e. $y = x+e$. The syndrome s is defined by $s = yH^T = (x+e)H^T = eH^T$. The ith block of the received sequence, denoted by y_i, can be decoded correctly if we can determine e_i, the ith block of e. This determination is usually made by examining the syndrome s which contains all the available information about e while being independent of x. Since it is impractical to examine all of s, we are forced to restrict our attention to a segment of some given finite length. Because of the structure of H, this means that only a finite number of blocks of y, say $y_{i-\alpha}, \ldots, y_{i+\beta}$, make a nonzero contribution to the calculation of this segment of s. Therefore the syndrome
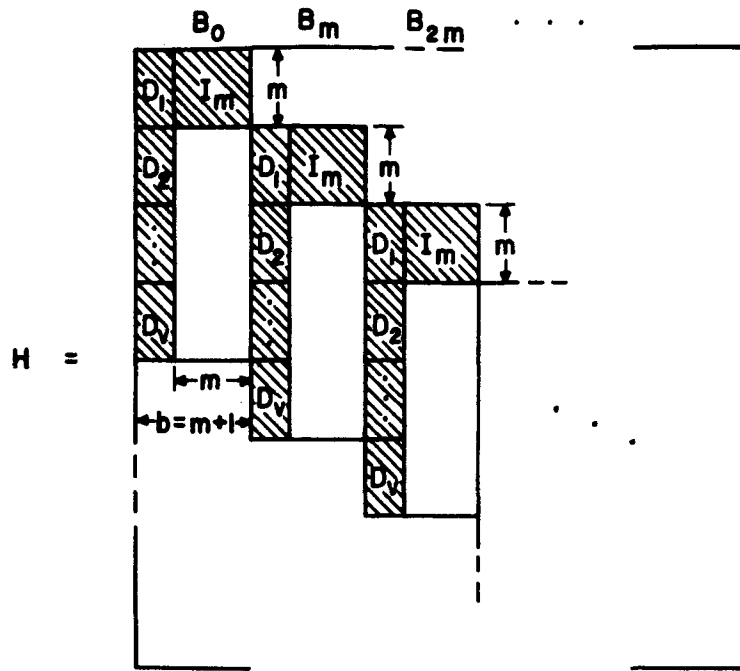
3

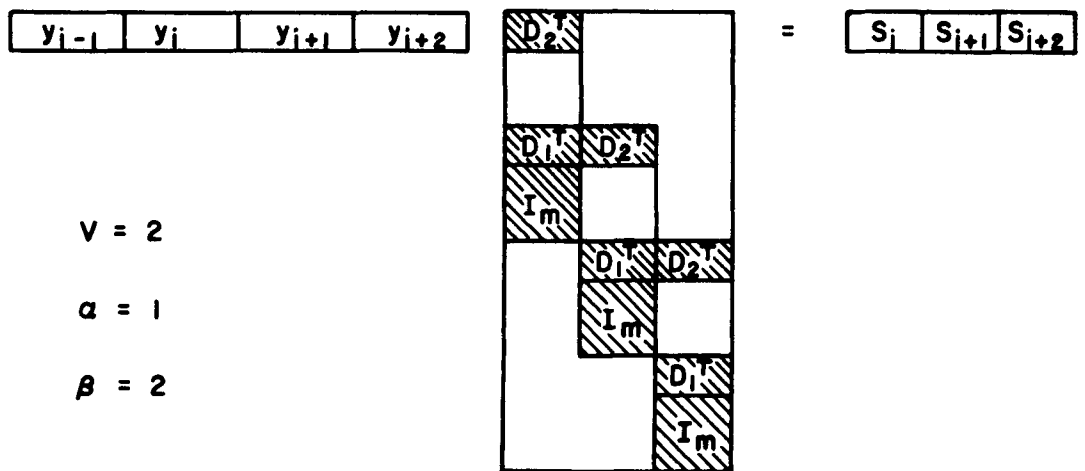Figure 1.   Parity check matrix of a rate 1/b systematic convolutional code.

Figure 2, Syndrome calculation for decoding y_i.

4

calculation required to decode y_i may be represented by a finite matrix

equation as illustrated in Figure 2 for $v = 2$, $\alpha = 1$, $\beta = 2$.

Let $\hat{e}_i$ denote the decoder's estimate of e_i. Then y_i is decoded

by subtracting $\hat{e}_i$ from y_i, i.e. $\hat{x}_i = y_i - \hat{e}_i$, where $\hat{x}_i$ denotes

the decoder's estimate of the ith transmitted block. (Normally, since

the code is assumed to be in systematic form, it is only necessary to

decode the information portion (first symbol) of block y_i. At this

point, however, we prefer to let this be understood rather than com-

plicate the notation.)

In the decoding process outlined above, no attempt is made to take

advantage of previous decoding successes. For example, in Figure 2

errors in block y_{i-1} can affect the syndrome calculation for block y_i

even though presumably e_{i-1} was correctly determined during the pre-

vious decoding step. It is therefore natural to consider substituting

$\hat{x}_j$ for y_j for some or all $j < i$. Robinson [10] has called the decoding

procedure with no substitution "definite decoding", and the decoding

procedure with substitution for all $j < i$ "feedback decoding".[*] If

we ignore the problem of error propagation in feedback decoders, the

only difference in the decoding problem using these two decoding options

is that the sets of correctable error patterns will not be the same.

Since the generalized one-step threshold decoding scheme discussed

here is applicable to any set of correctable error patterns, it will

---

[*]Apparently the intermediate case where substitution is made for some
but not all $j < i$ has not been considered. It might be interesting
to investigate the relationship between decoding constraint length
and error propagation in this more general case.

5

not be necessary to distinguish between definite and feedback decoding in the mathematical development to follow.

We will now use H, y, e and s to denote the finite submatrices used in the decoding of y_i. Thus the equation of Figure 2 is written simply as $yH^T = s$. This should cause no confusion since henceforth we will not have occasion to use these symbols in their infinite sense.

When definite decoding is used, $s = yH^T$. However when feedback decoding is used, the vector $y = (y_{i-\alpha}, \ldots, y_{i+\beta})$ is replaced by the vector $(\hat{x}_{i-\alpha}, \ldots, \hat{x}_{i-1}, y_i, \ldots, y_{i+\beta})$. It will be convenient to let $\bar{y}$ denote either of these vectors.

When definite decoding is used, the relationship between syndromes and coset leaders is the same as in the decoding of linear block codes.[*] That is, a minimum weight vector from each coset is selected to be the coset leader, thus establishing a 1-1 correspondence between coset leaders and the p^r distinct syndromes, where $r = (\alpha+\beta - V + 2)m$. When feedback decoding is used, the coset leader is a vector which has minimum weight among those members of the coset which have no nonzero components in blocks $i-\alpha, \ldots, i-1$. In either case, let $\hat{e}_i(s)$ denote the ith block of the coset leader associated with s. The code is then decoded up to its minimum distance with respect to the decoding constraint length (which is $(\alpha+\beta+1)b$ for definite decoding, $(\beta+1)b$ for feedback decoding) by

---

[*]The reader not familiar with the elementary properties of linear block codes is referred to Peterson [11, pp. 30-38].

6

$$\hat{x}_i = y_i - \hat{e}_i(s) \quad \text{where} \quad s = \bar{y}H^T.$$

The purpose of the next section is to show how this decoding procedure can be realized by a one-step threshold decoder.

III. Generalized Threshold Decoding Rule

In this section, all operations are performed over C, the field of complex numbers, unless otherwise indicated. Before proceeding, it is unfortunately necessary to establish more notation.

1) The symbol "$\dagger$" will be used to denote the complex exponentiation mapping $x^{\dagger} = \epsilon^x$, where $\epsilon = \exp(2\pi i/p)$. This mapping extends to matrices in the natural way, i.e. $(a_{ij})^{\dagger} = (a_{ij}^{\dagger})$.

2) Let A_r denote the p^r by p matrix whose rows are the p-ary numbers $0, 1, \ldots, p^r - 1$ in that order. A_r is a representation of $V_r(J_p)$, the vector space of r-tuples over J_p, the integers modulo p. Let f denote a p^r by 1 vector with components from C. A_r and f together represent, in tabular form, a function from $V_r(J_p)$ into .

3) We define a threshold function T as follows. Let $x = \rho \epsilon^{\Theta}$ be any complex number, where ρ and Θ are real numbers in the range $\rho \geq 0$ and $-1/2 \leq \Theta < p - 1/2$, and $\epsilon = \exp(2\pi i/p)$. Then $T(x) = [\Theta + 1/2]$, where the square brackets denote "integer part of". (That is, we threshold on the angle of the complex number x. In the binary case, this amounts to thresholding on the sign of $Re(x)$.) Note the following properties of T:

    a) If x is real, $T(x^{\dagger}) \equiv [x + 1/2] \pmod{p}$.

    b) If x is an integer, $T(x^{\dagger}) \equiv x \pmod{p}$.

    c) If x is any complex number and y is an integer, then
        $y + T(x) \equiv T(y^{\dagger}x) \pmod{p}$.

8

4) Some matrix notation that will be used is:

    a) I_n is the identity matrix of order n.

    b) The transpose and conjugate transpose of a matrix A will be denoted by A^T and A^* respectively.

    c) The scalar (element-wise) product of two matrices A and B will be denoted by $A \circ B$.

We now have

Theorem 1

Any function f from V_r (J_p) into C can be expressed as the Fourier Series

$$f(z) = (zA_r^T)^\dagger w_o \quad \text{where} \quad w_o = p^{-r}((A_r A_r^T)^\dagger)^* f.$$

Proof

The proof of this theorem will only be sketched here. For details, the reader is referred to [7]. First, it is established that $p^{-\frac{r}{2}}(A_r A_r^T)^\dagger$ is a unitary matrix. (A matrix M is unitary if $MM^* = I$.) Then we have

$$f = f$$

$$= I_{p^r} f$$

$$= p^{-r}(A_r A_r^T)^\dagger ((A_r A_r^T)^\dagger)^* f.$$

Let $w_o = p^{-r}((A_r A_r^T)^\dagger)^* f$. ($w_o$ is the p-point, r-dimensional finite Fourier transform [12] of f and will be referred to as the "spectrum"

9

of f.) Then $f = (A_r A_r^T)^\dagger w_o$. Since the rows of A_r constitute the complete set of possible input vectors z, this can be written in the functional form $f(z) = (z A_r^T)^\dagger w_o$.   Q.E.D.[*]

Corollary 1

For every function f from $V_r(J_p)$ into $J(p)$, there exists a vector w such that

$$f(z) = T((z A_r^T)^\dagger w).$$

Proof

We know from Theorem 1 that $f^\dagger(z)$ can be expressed as the Fourier series $f^\dagger(z) = (xA_r^T)^\dagger w_o$, where $w_o = p^{-r}((A_r A_r^T)^\dagger)^* f^\dagger$. The assertion then follows by observing that $f(z) = T(f^\dagger(z))$ by property (b) of T.   Q.E.D.

The spectrum w_o is a canomical weight vector that always satisfies the equation. Clearly there is an infinite convex set of weight vectors that satisfy the equation. The major problem in the synthesis of threshold decoders is to find a weight vector with the minimum number of nonzero components. As we shall see, this yields a decoder which employs the minimum number of estimators.

At this point, it is convenient to introduce notation for the information portion of a block. Let y_{ij}, $j = 0,1,\ldots,b-1$, denote the symbols of block y_i. Then y_{io} is the information portion (first symbol) of block y_i

---

[*]This result may also be obtained from the fact that $(A_r A_r^T)^\dagger$ is the character table of the Abelian group whose elements are the rows of A_r under modulo p addition. The characters of an Abelian group form an orthogonal basis for all complex-valued functions defined on the group [13].

and similarly for x_i and e_i. We now show the existence of a one-step threshold decoding function for a rate 1/b systematic convolutional code. Let B denote the matrix obtained by the modulo p addition of 1 to every element of the first column of the ith block of $A_r H$. Then we have

Theorem 2

For every rate 1/b systematic convolutional code over GF(p), there exists a vector w such that the decoding function

$$\hat{x}_i = T((\bar{y} \; B^T)^\dagger w)$$

decodes the code up to its minimum distance with respect to the decoding constraint length.

Proof

Let f be the p^r by 1 vector whose s^{th} component is $- \hat{e}_{io}(s)$. The syndromes, ordered lexicographically, form the rows of A_r. Then from corollary 1 we know that the function f can be expressed as

$$f(s) = - \hat{e}_{io}(s) = T((s \; A_r^T)^\dagger w).$$

From Section I, we have that a convolutional code can be decoded up to its minimum distance with respect to the constraint length by the decoding rule

$$\hat{x}_{io} \equiv y_{io} - \hat{e}_{io}(s) \pmod{p}$$

11

Combining these two equations, we have

$$\hat{x}_{io} \equiv y_{io} + T((s\ A_r^{\ T})^{\dagger} w) \pmod{p}.$$

Up to this point, we have viewed $s\ A_r^{\ T}$ as a product over C. However, s and $A_r^{\ T}$ both have components from $V_r(J_p)$ and the exponentiation operator "$\dagger$" has the effect of reducing its operand modulo p. It follows that we may, if we choose, view $s\ A_r^{\ T}$ as a product over J_p. But then

$$s \equiv \bar{y}\ H^T \pmod{p}$$

may be substituted into the decoding equation to give

$$\hat{x}_{io} \equiv y_{io} + T((\bar{y}\ H^T\ A_r^{\ T})^{\dagger} w) \pmod{p}.$$

Now since y_{io} is an integer, we can use property (c) of T to rewrite this as

$$\hat{x}_{io} = T((y_{io}^{\dagger} \circ (\bar{y}\ H^T\ A_r^{\ T})^{\dagger} w).$$

Then this can be rearranged (using the fact that for any matrices A and B, $A^{\dagger} \circ B^{\dagger} = (A + B)^{\dagger}$; see [7] for details) to give the desired result

$$\hat{x}_{io} = T((\bar{y}\ B^T)^{\dagger} w),$$

where B is obtained by adding (modulo p) 1 to every component of the first column of the ith block of $A_r H$.   Q.E.D.

12

Schematic diagrams of the definite and feedback decoder configurations described by $\hat{x}_{io} = T((\bar{y}\ B^T)^{\dagger}\ w)$ are shown in Figure 3 for $\alpha = 1$, $\beta = 2$. The matrix B and weight vector w are shown primed in Figure 3(b) to indicate that the decoding logic for a given convolutional code is, in general, different for the definite and feedback decoding options.

The existence of a one-step threshold decoder for a rate 1/b systematic code having been established, we now turn to the practical problem of decoder synthesis. Once the finite parity check matrix H has been specified, and it has been decided whether or not feedback will be used, the only variable in the design of a one-step threshold decoder is the weight vector w. So the decoder synthesis problem boils down to finding a vector w that satisfies $-\hat{e}_{io}(s) = T((s\ A_r^T)^{\dagger}\ w)$ and has the minimum number of nonzero components. If we require that w satisfy the above constraint for all s (as we have implicitly assumed up to now), then the decoder is roughly analogous to a maximum likelihood decoder for block codes. In practice, however, it is usually only required that w satisfy the constraint for all s that correspond to coset leaders of weight t or less, where t is the guaranteed error-correction capability of the code. In the examples to follow, decoders are designed to meet only this latter requirement. (This is not to say that some error patterns of weight greater than t will not be corrected; it is simply that we do not specify which patterns these shall be.)

13
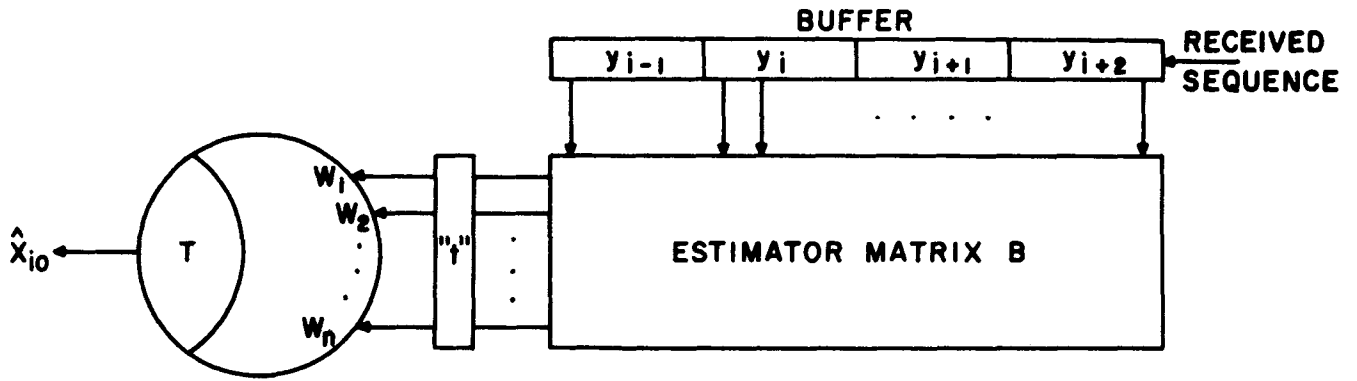
(a) Definite decoding
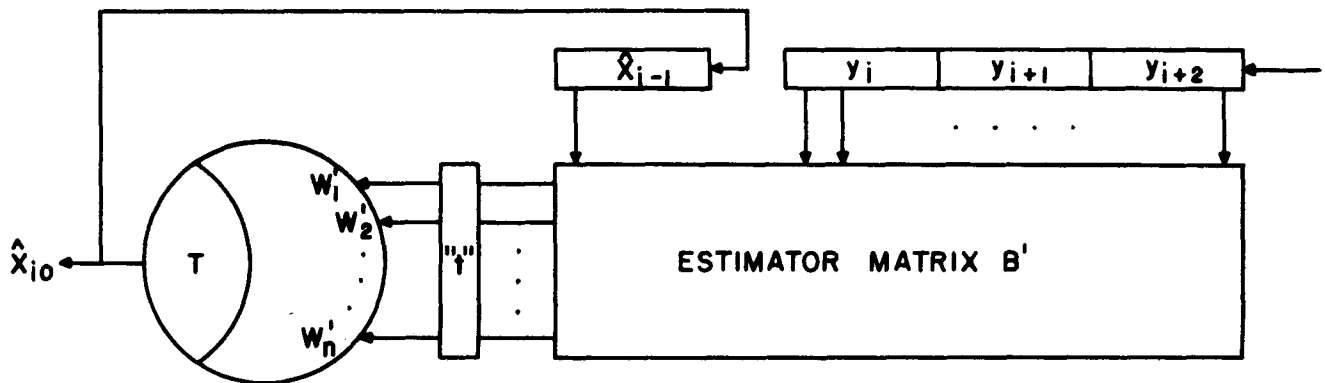
(b) Feedback decoding

Figure 3. One-step threshold decoding configurations.

To date, no efficient algorithm has been devised for finding an optimal weight vector w. However, good results have been obtained by starting with the spectrum w_o and deleting Fourier coefficients which are small in absolute value while at the same time readjusting the larger coefficients in order to stay within the convex set of solutions. Some examples will illustrate the approach.

Example 1

Take $\alpha = \beta = 1$, where $B_o = \begin{bmatrix} 11 \\ 10 \end{bmatrix}$. Then the parity check matrix involved in the decoding of y_{io} is

$$H = \begin{bmatrix} 10 & \overbrace{11}^{i^{th} \text{ block}} & 00 \\ 00 & 10 & 11 \end{bmatrix}.$$

This is the parity check matrix of a rate 1/2 binary self-orthogonal code [2]. Consider first the synthesis of a definite decoder.

For this code, $t = 1$ so we need only consider those syndromes associated with coset leaders of weight 1 or less (in this case, all of them). The relationship between syndromes and coset leaders is shown in Table I.

coset leader	syndrome
00 00 00	00
00 00 01	01
00 01 00	10
00 $\underbrace{10 \quad 00}_{\hat{e}_i}$	11

Table I.

15

Then

$$f = - \hat{e}_{io}(s) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} .$$

The spectrum of $f^{\dagger}$ is

$$w_o = p^{-r}((A_r A_r^T)^{\dagger})^* f^{\dagger} = 2^{-2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ -1/2 \end{bmatrix} .$$

The matrix $A_r H$ for this code is

$$A_2 H = \begin{bmatrix} 00 \\ 01 \\ 10 \\ 11 \end{bmatrix} \begin{bmatrix} 10 & 11 & 00 \\ 00 & 10 & 11 \end{bmatrix} = \begin{bmatrix} 00 & 00 & 00 \\ 00 & 10 & 11 \\ 10 & 11 & 00 \\ 10 & 01 & 11 \end{bmatrix} .$$

The estimator matrix B is obtained by adding 1 (mod 2) to each element of the first column of the ith block of $A_2 H$. This gives

$$B = \begin{bmatrix} 00 & 10 & 00 \\ 00 & 00 & 11 \\ 10 & 01 & 00 \\ 10 & 11 & 11 \end{bmatrix} .$$

Then a decoding function for this code is

$$\hat{x}_{io} = T((\bar{y}\ B^T)^{|}\ w_o)$$

$$= T\left(\left((y_{i-1,0}, u_{i-1,1}, y_{i0}, y_{i1}, y_{i+1,0}, y_{i+1,1})\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}\right)^{+}\begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ -1/2 \end{bmatrix}\right)$$

It is easy to verify that no weight vector with fewer than three nonzero components can be a solution to $-\hat{e}_{io}(s) = T((s\ A_r^T)^{+}w)$. Since the moduli of the components of the spectrum w_o are all equal in this case, we suspect that a minimal solution may be found whose three components are equal in absolute value and have the same signs as the corresponding spectral components. (We conjecture that an optimal solution w can always be found such that its nonzero components agree in sign with corresponding components of w_o.) Our suspicions are justified in this case; one optimal solution is

$$w = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

This yields the reduced decoding function

$$\hat{x}_{io} = T\left(\left((y_{i-1,0}, \cdots, y_{i+1,1})\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}\right)^{+}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\right)$$

17

where we have deleted the last column of B^T and the corresponding

(zero) component of w. The corresponding decoder is shown in

Figure 4(a). Note that the estimators are disjoint here (as might

have been expected since this code is orthogonalizable).

Another minimal weight vector is

$$w = \begin{bmatrix} 0 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

and the decoding function for this choice of w is

$$\hat{x}_{io} = T\left(\left((y_{i-1,0},\cdots,y_{i+1,1})\begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}\right)^{\dagger}\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}\right) \; .$$

The corresponding decoder is shown in Figure 4(b). In this case the

estimators are not disjoint. Both decoders are capable of correcting

any single error in a definite decoding constraint length of $(\alpha + \beta + 1)b = 6$

bits. The aesthetic choice is decoder (a), of course, but decoder (b)

does illustrate the fact that an estimator that is consistently wrong

is just as useful as one that is consistently right; we simply comple-

ment the estimate (weight it by -1) in the former case.

Now consider the design of a feedback decoder for this code. In

this case, we want the coset leaders to be minimum weight vectors with

no nonzero components in the (i-1)st block. But the coset leaders
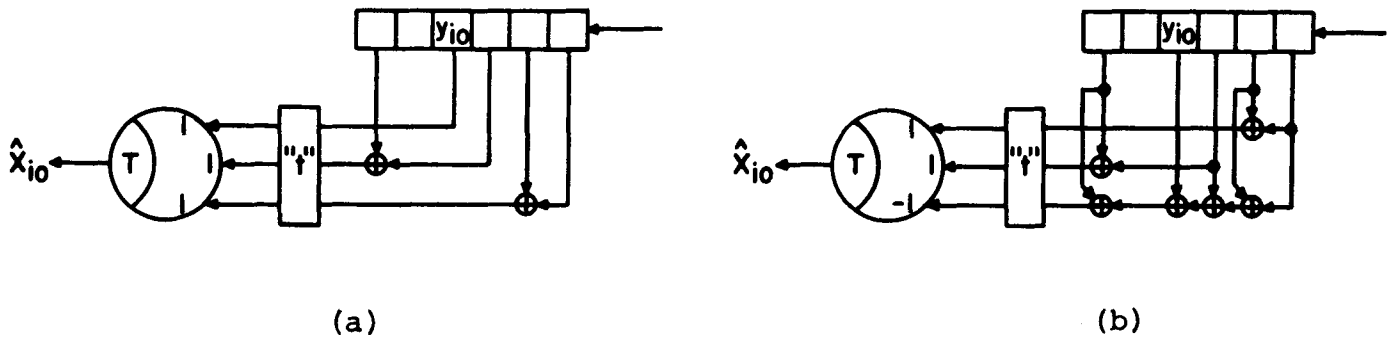
18

NOTE: ⊕ DENOTES MOD 2 ADDER

(a)

(b)

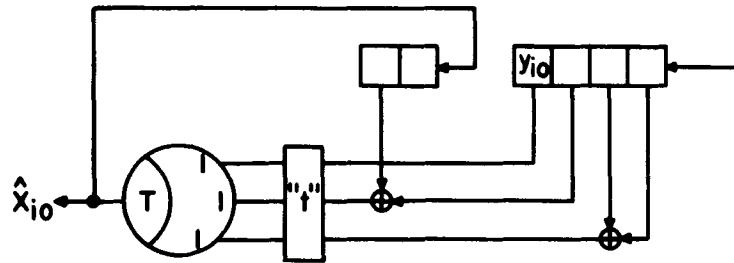Figure 4. Definite decoders for b=2, m=1, v=2 self-orthogonal code.

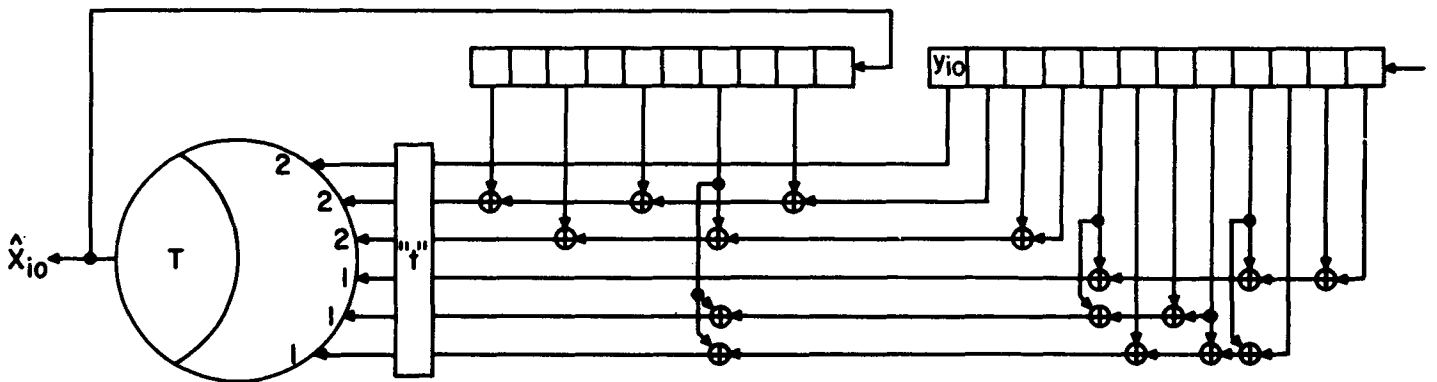Figure 5. Feedback decoder corresponding to figure 4(a).

Figure 6. Feedback decoder for b=2, m=1, v=6 code.

listed in Table I already have that property. Therefore the feedback decoding logic in this case is exactly the same as the definite decoding logic. The feedback decoder corresponding to the definite decoder of Figure 4(a) is shown in Figure 5. The decoding constraint length here is $(\beta+1)b = 4$ bits.

The reader may have noticed that in the binary case the exponentiation operation "$+$" could be eliminated by a simple modification of the threshold operator. (For example, the exponentiation and threshold operators in Figure 4(a) can be replaced by a simple majority element.) This is because in the binary case $x^{+} = 1-2x$, which is a trivial mapping. In the nonbinary case, however, the exponentiation mapping is no longer trivial, and we prefer to leave things as they are for the sake of uniformity. Also, as we shall see in the next section, exponentiation becomes nontrivial even in the binary case when we consider the extension to analog threshold decoding.

The example just worked is of limited interest to the designer because the code is orthogonalizable and we already know that a simply majority decoder exists [2]. Therefore we now exhibit a threshold decoder for a code that is not orthogonalizable.

Example 2

Take $\beta = \alpha = 5$, where

$$B_o = \begin{bmatrix} 11 \\ 10 \\ 00 \\ 10 \\ 00 \\ 10 \end{bmatrix}$$

20

Then

$$H = \begin{bmatrix} 10 & 00 & 10 & 00 & 10 & 11 & & & & & \\ & 10 & 00 & 10 & 00 & 10 & 11 & & & & \\ & & 10 & 00 & 10 & 00 & 10 & 11 & & & \\ & & & 10 & 00 & 10 & 00 & 10 & 11 & & \\ & & & & 10 & 00 & 10 & 00 & 10 & 11 & \\ & & & & & 10 & 00 & 10 & 00 & 10 & 11 \end{bmatrix} .$$

This is a parity check matrix (taken from Bussgang [14]) of a rate 1/2 binary code capable of correcting all single and double errors in a feedback decoding constraint length of 12 bits. Employing the same general approach used in Example 1, the following (probably minimal) decoding function was found.

$$\hat{x}_{io} = T\left(\left((\hat{x}_{i-5,0}, \cdots, y_{i+5,1}) \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \right)^+ \begin{bmatrix} 2 \\ 2 \\ 2 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right) .$$

21

The corresponding decoder is shown in Figure 6. The estimators in this case are not disjoint, as must be the case when the code is not orthogonalizable.

IV. Analog Threshold Decoding

In the previous section, it was convenient to view the matrix product $\bar{y} B^T$ in $\hat{x}_{io} = T((\bar{y} B^T)^\dagger w)$ as an operation over $GF(p)$. This leads quite naturally to a decoder implementation using a shift register, mod p adders and multipliers, an exponentiation circuit, and a linear threshold gate. The only nonstandard item is the exponentiation circuit, and it was pointed out that this can be eliminated in the binary case.

Suppose, however, we choose to view $\bar{y} B^T$ as a product over C and drop the restriction that the components of $\bar{y}$ be integral. Then $\hat{x}_{io} = T((\bar{y} B^T)^\dagger w)$ represents a continuous extension of the original discrete decoding function, and a physical device that realizes this continuous function can "decode" a received sequence in its unquantized analog form. We now consider two possible methods of implementing such a device.

The first method employ dc devices only, and in this case we prefer to restrict our attention to the binary case so that we are dealing with real quantities throughout. (We are assuming that $\bar{y}$ is real.) As pointed out earlier, the operator T thresholds on the sign of the real part of its argument in the binary case. Hence there is no objection to replacing the argument by its real part. Then the decoding function can be written $\hat{x}_{io} = T(\text{Re}((\bar{y}B^T)^\dagger w))$. We can take w to be real in the binary case (w_o is real), so we need only replace

23

$(\bar{y} \ B^T)^\dagger$ by its real part. The exponentiation operator in the binary

case is $z^\dagger = -1^z = \cos \pi z - i \sin \pi z$. Hence Re $(\bar{y} \ B^T)^\dagger = \cos \pi(\bar{y} \ B^T)$,

where the cosine function is extended to matrices in the natural way.

Then the decoding function can be written as

$$\hat{x}_{io} = T((\cos \pi(\bar{y} \ B^T)) \ w).$$

The product $\bar{y} \ B^T$ over the reals is a vector whose ith component

is the correlation between $\bar{y}$ and the ith column of B^T. Various

standard devices are available to perform this operation, e.g. a

tapped delay line and linear summing amplifiers. The cosine operator can

be realized by any device with a sinusoidal response function, e.g. a

phase modulator-demodulator pair. The last component is a conventional

linear threshold gate. The resulting analog decoder configuration is

shown in Figure 7.

The second method of implementing the extended decoding function

makes use of both dc and microwave devices. In this case complex

quantities cause no problems, so we need no longer restrict our atten-

tion to the binary case. The vector $(\bar{y} \ B^T)^\dagger$ describes the outputs

of a phase shifter matrix where the microwave inputs are all carriers

at a common carrier frequency with unit amplitude and 0° relative

phase (provided by an rf source and a stripline feed structure), the

components of $\bar{y}$ are the dc control inputs, and the coupling between

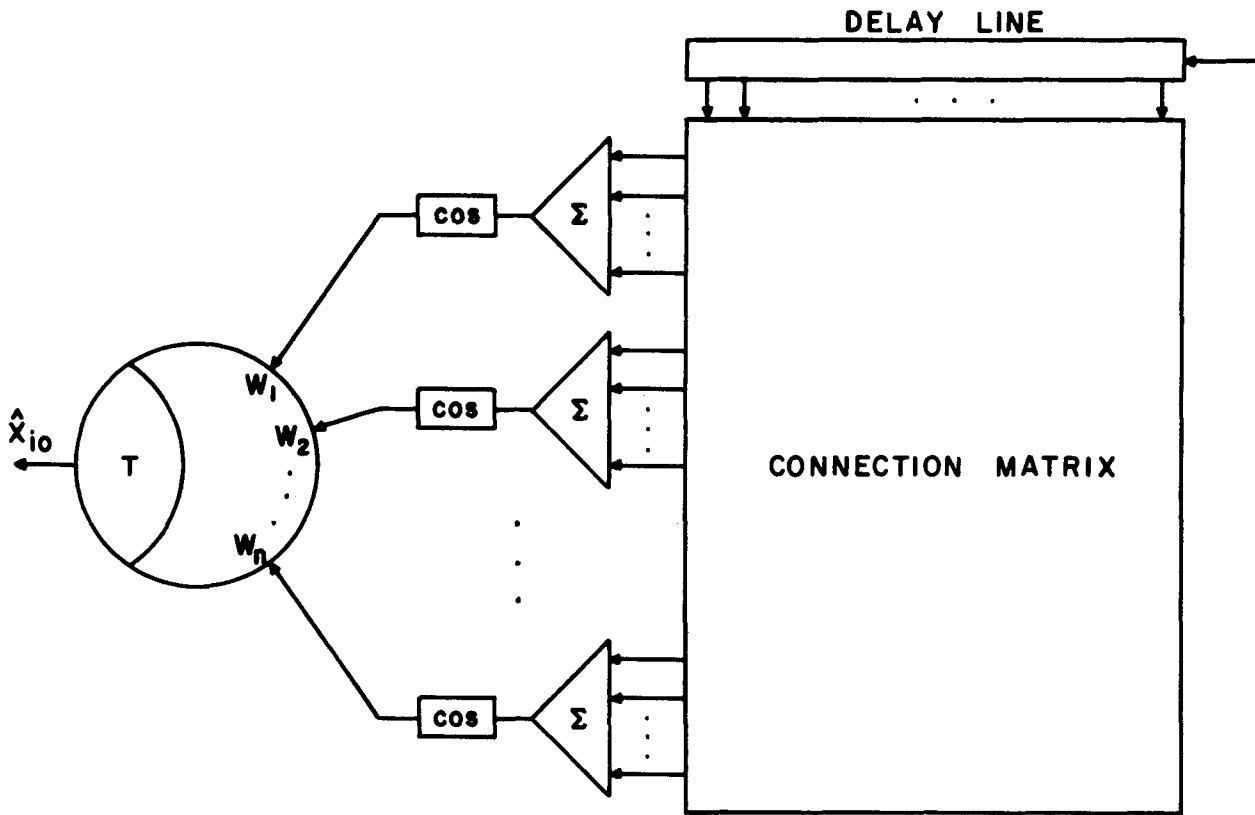dc control lines and the microwave lines is specified by the matrix B.
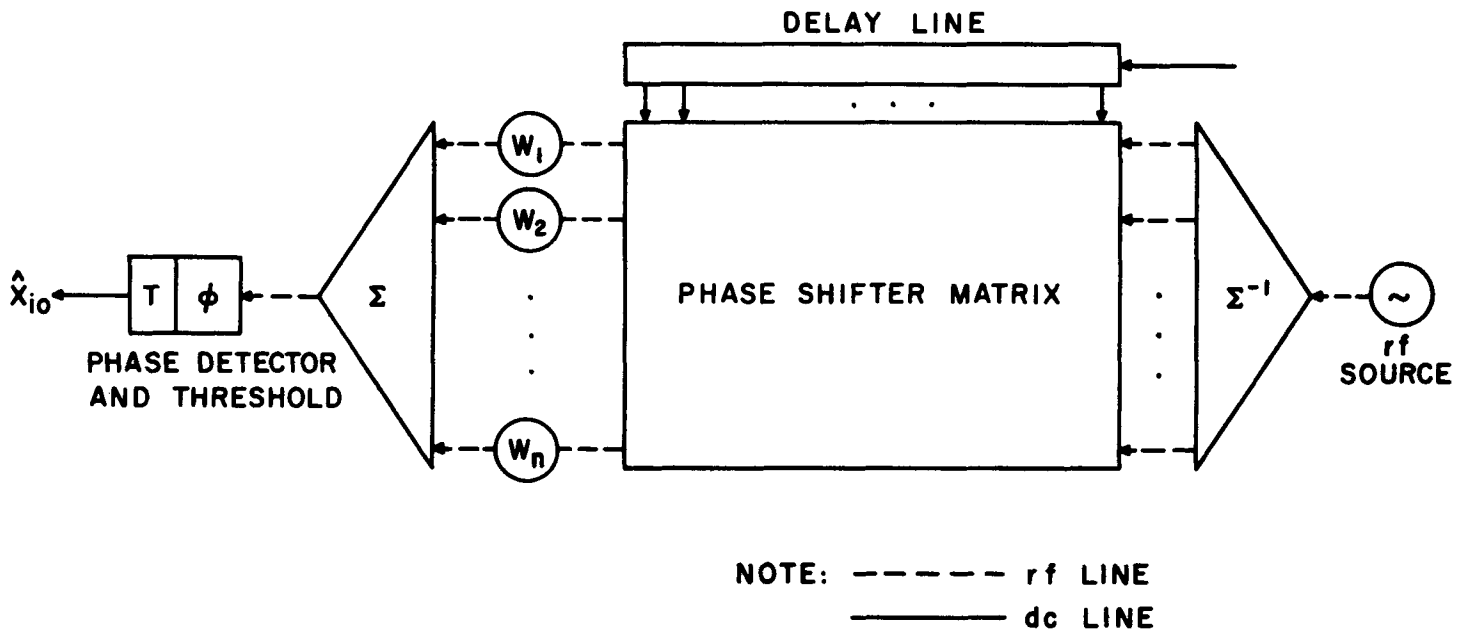
24

Figure 7. dc analog threshold decoder (binary only).

NOTE: $-----$ rf LINE
$\underline{}$ dc LINE

Figure 8. rf-dc analog threshold decoder.

The complex scalar $(\bar{y}\, B^T)^\dagger\, w$ is a weighted sum of the outputs of the phase shifter matrix. The weights are achieved by fixed attenuation and phase shift, and the sum formed in a second reciprocal feed structure. Finally, $T((\bar{y}\, B^T)^\dagger\, w)$ is obtained by adding a phase detector and thresholding device at the output. The resulting analog decoder is shown in Figure 8. (See [15] for further discussion.)

We make no claims for the implementation options proposed above. The main reason for including a discussion of hardware is to provide a "physical picture" of the decoding function $\hat{x}_{io} = T((\bar{y}\, B^T)^\dagger\, w)$.

IV. Concluding Remarks

It has been shown that any rate 1/b systematic convolutional code over GF(p) can be decoded up to its minimum distance with respect to the decoding constraint length by a one-step threshold decoder. The major problem in the synthesis of these decoders is to find an efficient algorithm for obtaining a weight vector with the minimum number of nonzero components. At this point, we do not even have a decent bound on the number of nonzero components that might be required.

The decoding function $\hat{x}_{io} = T((\bar{y} \, B^T)^+ \, w)$ can be viewed as a description of a digital threshold decoder which, in the binary case anyway, can be implemented with conventional logical components. One possible advantage of the decoder configuration presented here is that it is not required to be block-synchronized with the received sequence and therefore might permit faster recovery from a synchronization error. On the other hand, the decoder operates at the received symbol rate rather than at the block rate which could be a disadvantage in some applications.

The decoding function $\hat{x}_{io} = T((\bar{y} \, B^T)^+ w)$ can also be viewed as a description of an analog threshold device that "decodes" the received sequence y in its unquantized analog form. Eliminating the quantizing step (or rather moving it from the decoder input to decoder output) should result in improved performance. In fact there is some reason to believe that this particular continuous extension of the discrete decoding function might be optimal in some sense. However, we make no claims at this time.

REFERENCES

[1] I.S. Reed, "A Class of Multiple-error-correcting Codes and the Decoding Scheme", <u>IRE Trans. on Information Theory</u>, <u>IT-4</u>, pp. 38-49, 1954.

[2] J.L. Massey, <u>Threshold Decoding</u>, M.I.T. Press, 1963.

[3] M.E. Mitchell et al., "Coding and Decoding Operations Research," Final report on Contract (AFCRL)AF 19(604)-6183, General Electric Co., 1961.

[4] L.D. Rudolph, "Geometric Configurations and Majority Logic Decodable Codes," M.E.E. Thesis, University of Oklahoma, 1964.

[5] E.J. Weldon, Jr., "Some Results on Majority Logic Decoding," <u>Proc. of Symposium on Error Correcting Codes</u>, University of Wisconsin (MRC), May 1968.

[6] W.C. Gore, "Generalized Threshold Decoding of Linear Codes," <u>IEEE Trans. on Information Theory, IT-15</u>, September, 1969.

[7] L.D. Rudolph, "Threshold Decoding of Cyclic Codes," <u>IEEE Trans. on Information Theory, IT-15</u>, pp. 414-418, May 1969.

[8] A.D. Wyner and R.B. Ash, "Analysis of Recurrent Codes," <u>IEEE Trans. on Information Theory, IT-9</u>, pp. 143-156, July 1963.

[9] A.D. Wyner, "On the Equivalence of Two convolution Code Definitions," <u>IEEE Trans. on Information Theory</u> (Corresp.), <u>IT-11</u>, pp. 600-602, October 1965.

[10] J.P. Robinson, "Error Propagation and Definite Decoding of Convolutional Codes," <u>IEEE Trans. on Information Theory</u>, <u>IT-14</u>, pp. 121-128, January 1968.

[11] W.W. Peterson, <u>Error-Correcting Codes</u>, M.I.T. Press, 1961.

[12] I.J. Good, "The Interaction Algorithm and Practical Fourier Analysis," <u>J. Roy. Stat. Soc., Vol. 20(2), Ser. B</u>, 1958.

[13] H. Rademacher, <u>Lectures on Elementary Number Theory</u>, Blaisdell, 1964.

[14] J.J. Bussgang, "Some Properties of Binary Convolutional Code Generators," IEEE Trans. on Information Theory, IT-11, pp. 90-100, January 1965.

[15] L. Rudolph, P. Blackwell and D. Shurtleff, "Data Processing Using Distributed Parameter Techniques," Reports 1,2,F on Contract (ECOM) DAAB07-67-C-0223, Syracuse University Research Corporation, 1968.