9

The Fourier Convolution Theorem over Finite Fields: Extensions of Its Application to Error Control Coding

Eric Sakk and Schinnel Small
Department of Computer Science, Morgan State University, Baltimore, MD, USA

1. Introduction

Linear spectral transform techniques such as the discrete Fourier transform and wavelet analysis over real and complex fields have been routinely applied in the literature (Burrus et al. (1998); Strang & Nuygen (1996)). Furthermore, extensions of these techniques over finite fields (Blahut & Burrus (1991); Caire et al. (1993)) have led to applications in the areas of information theory and error control coding (Blahut (2003); Dodd (2003); Sakk (2002); Wicker (1994)). The goal of this chapter is to review the Galois Field Fourier Transform, the associated convolution theorem and its application in the field of error control coding. In doing so, an interesting connection will be established relating the convolution theorem over finite fields to error control codes designed using finite geometries (Blahut (2003); Lin & Costello (1983); Wicker (1994)).

While a complete exposition of the field of error control would be out of context for this chapter, we refer the interested reader to the recent characterizations of Low-Density Parity Check (LDPC) codes (Pusane et al. (2011); Smarandache et al. (2009); Xia & Fu (2008)). Such formulations have led to a resurgence of interest in the design (Kou et al. (2001); O.Vontobel et al. (2005); Tang et al. (2005); Vandendriesscher (2010)) and decoding (Kou et al. (2001); Li et al. (2010); Liu & Pados (2005); Ngatched et al. (2009); Tang et al. (2005); Zhang et al. (2010)) of finite geometry codes. The formulation in this chapter is meant to serve as a guiding principle relating finite geometric properties to algebraic ones. The vehicle we have chosen to demonstrate these relationships is an example from the field of error control. In particular, we show how a generalized Fourier-like convolution theorem can be applied as a decoding methodology for finite geometry codes.

We begin in Section 2 by reviewing the Galois Field Fourier Transform (GFFT) followed by an overview of error control coding in Section 3. In addition, in Section 3.1 it is demonstrated how the GFFT can be applied within the context of error control coding. Section 4 then goes on to generalize these results to linear transformations using Pascal's triangle as an example. The combinatorics of such a transformation naturally lead to the design of codes derivable from

finite geometries. Finally, Sections 5 and 6 conclude this chapter by deriving and applying the generalized convolution theorem.

2. The Galois Field Fourier Transform

We are particulary interested in the case of finite fields where p is a prime number and $\alpha \in GF(p^m)$ is an element of order n. The Galois Field Fourier Transform (GFFT) and its inverse of a vector $v = \{v_0, v_1, ..., v_{n-1}\}$ over $GF(p)$ of length n can be related via the equations:

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i \qquad j = 0, ..., n-1$$

and

$$v_i = (n)^{-1} \sum_{j=0}^{n-1} \alpha^{-ij} V_j \qquad i = 0, ..., n-1.$$

For any vector f over $GF(p)$ where the above equations hold true, we define

$$\mathcal{F}(v) \equiv V = \{V_0, V_1, ..., V_{n-1}\} \tag{1}$$

as the GFFT of v and

$$\mathcal{F}^{-1}(V) = v = \{v_0, v_1, ..., v_{n-1}\} \tag{2}$$

as the inverse GFFT of F.

Using this formulation, given two vectors

$$\begin{aligned} v &= \{v_0, v_1, ..., v_{n-1}\} \\ w &= \{w_0, w_1, ..., w_{n-1}\} \end{aligned} \tag{3}$$

over $GF(p)$ and their associated transforms

$$\begin{aligned} \mathcal{F}(v) &= V = \{V_0, V_1, ..., V_{n-1}\} \\ \mathcal{F}(w) &= W = \{W_0, W_1, ..., W_{n-1}\}, \end{aligned} \tag{4}$$

the familiar convolution theorem can be demonstrated to hold true for the finite field case. Specifically, computing

$$x_j = \sum_{k=0}^{n-1} v_k w_{(j-k)} \tag{5}$$

is equivalent to computing

$$x_j = \mathcal{F}^{-1}(V_j W_j). \tag{6}$$

3. Error control coding

Given a message encoded as a vector μ of length k over $GF(p)$, the goal of error control coding (ECC) is to transform the message vector into a code vector C of length $n > k$ in a way that causes C to be robust to errors arising over a communication channel (such as a wireless

link, fiber optic cable, etc). Rather than the message vector μ, it is the code vector C that is transmitted over a channel where the receiver is only able to observe a received vector $\hat{C}$. Ideally, in the absence of any noise, it should be the case that $\hat{C} = C$. On the other hand, if noise is present on the channel, the method used to transform (i.e. 'encode') the message μ into the code vector C provides a way to recover μ from $\hat{C}$. The basic strategy behind ECC is, given a message,

a. Embed a k dimensional message vector μ in a larger vector space of dimension n to create the code vector C.
b. The addition of channel noise converts C into the received vector $\hat{C}$.
c. If the channel noise does not cause $\hat{C}$ to be confused with other possible encodings, the original code vector C can be recovered using some predetermined decoding scheme. Conceptually speaking, the $\hat{C}$ that lies within a predefined noise 'sphere' with respect to the original C will be decoded as the (ideally) unique C; hence, μ can be recovered as well. The size of the noise sphere (which is designed as part of the code) determines how many errors can be corrected.

The general idea behind ECC then is to find a C that minimizes $||C - \hat{C}||$; however, numerically determining the minimum distance solution is wrought with dimensionality issues that can lead to computational intractability. Hence, classes of codes have been devised that relate the message encoding method to the decoding algorithm. Such algorithms are often iterative (Blahut (2003); Lin & Costello (1983); Wicker & Kim (2003)) and converge upon the optimal solution by exploiting the mathematical structure designed into the code.

Two important quantities in the field of ECC are the Hamming weight and the Hamming distance. Consider two vectors v and w of length n over $GF(p)$.

Definition 3.1. *The Hamming weight $w_H(v)$ of a vector v is defined as the number of non-zero components in v.*

Definition 3.2. *The Hamming distance between v and w is defined as the number of components that differ between v and w.*

For example, over $GF(3)$, assuming $n = 5$, $v = \{0\ 2\ 1\ 0\ 2\}$ and $w = \{0\ 2\ 2\ 1\ 2\}$, according to the above definitions we have that $w_H(v) = 3$, $w_H(w) = 4$ and $d_H(v, w) = 2$.

An important quantity for defining the noise sphere is referred to as d_{min} which is the minimum Hamming distance between all code vectors defined in the code class. To correct up to t errors in any code vector, it turns out that $d_{min} = 2t + 1$. Furthermore, when the ECC is a linear code, a major simplification arises where d_{min} is simply the minimum Hamming weight computed over all non-zero code vectors in the code class.

3.1 Application of the GFFT to Reed-Solomon codes

The GFFT and the convolution theorem have been applied in the field of error control coding for the construction of a class of linear codes known as Reed-Solomon codes (Blahut (2003); Wicker (1994)). The algorithm for encoding a message vector μ over $GF(p^m)$ of length k is

quite straightforward. To be able to correct up to t errors, create a vector of length n by appending μ with $2t$ consecutive zeros. The code vector C is then derived by computing the inverse GFFT of the appended construction. One approach to proving that this construction is capable of correcting up to t errors involves applying the GFFT convolution theorem. Specifically, given a code vector C, a *locator* vector Λ must be defined such that $C_j\Lambda_j = 0$ for all $j = 0, \cdots, n$. Letting c and λ denote the GFFT of C and Λ, the convolution theorem implies $c * \lambda = 0$. Based upon the convolution approach, the conclusion can be reached that the inverse GFFT construction leads to Reed-Solomon codes capable of correcting up to t errors in the code vector (Blahut (2003); Wicker (1994)).

The key feature of the GFFT approach to constructing Reed-Solomon codes described above is that restrictions are placed on the position and the number of zeros appended to the message vector. To summarize:

i. Addition of zeros to the message vector μ of length k is performed at prescribed locations.

ii. The resulting vector is then inverse transformed in order to compute the code vector C.

iii. The error correcting properties of this code can be demonstrated by applying the convolution theorem.

In this work, one our of goals is to demonstrate that, given other linear transformations inducing a convolution theorem, the above steps can be generalized to other classes of codes. As we shall see, the key is to define the transform and the structure of how zeros are introduced into the message vector.

4. Pascal codes

4.1 The Pascal matrix over finite fields

Let us now focus our attention on the case of $GF(p)$ where p is prime. Our starting point will be:

Definition 4.1. *Let p be a prime number, then the ij^{th} entry of a $p^m \times p^m$ $\boldsymbol{m^{th}}$ order Pascal matrix $\boldsymbol{P_{p^m}}$ over $\boldsymbol{GF(p)}$ is defined as*

$$\begin{aligned} p_{ij} &= (j!)((j-i)!i!)^{-1} \mod p \\ &= \binom{j}{i} \mod p \end{aligned} \tag{7}$$

for $i, j = 0,\ 1,\ ...,\ p^m - 1$ and, by convention, if $i > j$, then $p_{ij} = 0$.

In other words, P_{p^m} is an upper triangular matrix whose non-zero entries are the elements of Pascal's triangle taken *mod* p. For the purposes of this work, it is useful to observe that P_{p^m} also has a Kronecker product description (Sakk & Wicker (2003)):

$$P_{p^m} = P_p \otimes P_{p^{m-1}} \mod p \tag{8}$$

where P_p is a 1^{st} order Pascal matrix.

Example 4.2. *Consider the binary case where $p = 2$ and $m = 3$. Equation (8) gives*

$$P_{2^3} = \begin{bmatrix} 1&1&1&1&1&1&1&1 \\ 0&1&0&1&0&1&0&1 \\ 0&0&1&1&0&0&1&1 \\ 0&0&0&1&0&0&0&1 \\ 0&0&0&0&1&1&1&1 \\ 0&0&0&0&0&1&0&1 \\ 0&0&0&0&0&0&1&1 \\ 0&0&0&0&0&0&0&1 \end{bmatrix}.$$

Example 4.3. *Consider the ternary case where $p = 3$ and $m = 2$. Equation (8) gives*

$$P_{3^2} = \begin{bmatrix} 1&1&1&1&1&1&1&1&1 \\ 0&1&2&0&1&2&0&1&2 \\ 0&0&1&0&0&1&0&0&1 \\ 0&0&0&1&1&1&2&2&2 \\ 0&0&0&0&1&2&0&2&1 \\ 0&0&0&0&0&1&0&0&2 \\ 0&0&0&0&0&0&1&1&1 \\ 0&0&0&0&0&0&0&1&2 \\ 0&0&0&0&0&0&0&0&1 \end{bmatrix}.$$

4.2 The inverse of the Pascal matrix

As Section 5 will require understanding $Q_{p^m} \equiv P_{p^m}^{-1}$, we introduce the following.

Observation 4.4. *Let p be prime and let Q_p be the $p \times p$ matrix defined by*

$$q_{ij} = \begin{cases} (-1)^{j-i}\binom{j}{i} \mod p & \text{if } j \geq i, \\ 0 & \text{otherwise} \end{cases} \qquad \text{for i,j=0, 1, ..., p-1.} \tag{9}$$

Then $Q_p = P_p^{-1} \mod p$.

This result easily follows from the integer case (Call & Velleman (1993); Heller (1963)). Furthermore, it has been demonstrated that (Sakk (2002)):

Observation 4.5. *If p is prime and P_p is a 1^{st} order Pascal matrix over $GF(p)$, then*

$$P_p^p \ mod \ p = I_p \tag{10}$$

where $I_p = p \times p$ identity matrix.

Hence, it easily follows that

Corollary 4.6. *If p is prime and P_p is a Pascal matrix over $GF(p)$, then*

$$Q_p = P_p^{-1} \ mod \ p = P_p^{p-1} \ mod \ p. \tag{11}$$

Example 4.7. *A Pascal matrix over $GF(5)$ and its inverse:*

$$P_p = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 3 & 1 \\ 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q_p = P_p^4 = \begin{bmatrix} 1 & 4 & 1 & 4 & 1 \\ 0 & 1 & 3 & 3 & 1 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Based upon Equation (8), it should be clear that

$$Q_{p^m} = Q_p \otimes Q_{p^{m-1}} \; mod \; p. \tag{12}$$

Finally, based upon Equation (10), it also follows that, for the m^{th} order case,

$$P_{p^m}^p \; mod \; p = I_{p^m} \tag{13}$$

where $I_{p^m} = p^m \times p^m$ identity matrix. In a manner similar to the $m = 1$ case, this characterization provides a path to computing the m^{th} order inverse

$$Q_{p^m} \equiv P_{p^m}^{-1} \; mod \; p = P_{p^m}^{p-1} \; mod \; p. \tag{14}$$

4.3 Error control codes designed from Pascal matrices

In a manner similar to the GFFT approach to Reed-Solomon codes summarized in Section 3.1, it has been pointed out that P_{p^m} can also be used to transform message vectors with the appropriate coordinates set equal to zero (Sakk & Wicker (2003)). More precisely, we have the following:

Definition 4.8. *Consider an m^{th} order Pascal matrix over $GF(p)$ and let r be an integer such that $0 \leq r \leq m(p-1)$. Also, consider the p-ary expansion of an index*

$$i = i_0 p^0 + i_1 p^1 + \cdots + i_{m-1} p^{m-1}$$

*where $0 \leq i_j \leq p-1$ for $0 \leq j \leq m-1$. A codeword c for an **r^{th} order Pascal code of length p^m**, denoted by $P_p(r,m)$, is generated by*

$$C = \mu P_{p^m} \tag{15}$$

where

$$\mu = \begin{pmatrix} \mu_0 & \mu_1 & \cdots & \mu_{p^m-1} \end{pmatrix}$$

is a message vector of length $p^m - 1$ such that $\mu_i \in GF(p)$,

$$\begin{cases} \mu_i = 0 & if \quad w_p(i) > r \\ \mu_i \neq 0 & if \quad w_p(i) \leq r \end{cases} \tag{16}$$

and

$$w_p(i) \equiv \sum_{j=0}^{m-1} i_j.$$

Error control codes derived from the m^{th} order Pascal matrix over $GF(2)$ (i.e. binary data) have been related (Forney (1988); Massey et al. (1973)) to a class of codes known as r^{th} order binary Reed-Muller codes $RM(r,m)$ of length 2^m (MacWilliams & Sloane (1977); Wicker (1994)). In addition, it has been further demonstrated (Sakk (2002)) that $P_2(r,m)$ codes over $GF(2)$ are equivalent to $RM(r,m)$ codes with minimum distance $d_{min} = 2^{m-r}$. These observations have been extended where it has been demonstrated that $P_p(r,m)$ codes over $GF(p)$ are equivalent to generalized Reed-Muller codes (GRM) codes (Sakk (2002)).

To place this class of codes in the same context as that outlined in Section 3.1, we must show how to introduce zeros into the message vector, apply the Pascal matrix as the linear transformation and, based upon this transformation, introduce a convolution theorem. From the definition above, a given code is specified by choosing p, m and a value of $0 \leq r \leq m(p-1)$. The code vector length then becomes $n = p^m$; and, for this class of codes, a given value of r defines the length k of the message. The rest of the $n-k$ components of μ must be set to zero in a systematic way that leads to the minimum distance property of the code.

Example 4.9. *Consider P_{2^3} in Example 4.2 (hence, $n = 2^3 = 8$) and a message vector $\mu = (\mu_0,\ \mu_1,\ ...,\ \mu_7)$ and let s be the number of consecutive zeros in the vector μ for a given value of r:*

$$\begin{array}{lll}
r = 0\ (d_{min} = 8): & s = 7 & \mu = (\mu_0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0) \quad (k = 1) \\
r = 1\ (d_{min} = 4): & s = 3 & \mu = (\mu_0,\ \mu_1,\ \mu_2,\ 0,\ \mu_4,\ 0,\ 0,\ 0) \quad (k = 4) \\
r = 2\ (d_{min} = 2): & s = 1 & \mu = (\mu_0,\ \mu_1,\ \mu_2,\ \mu_3,\ \mu_4,\ \mu_5,\ \mu_6,\ 0) \quad (k = 7) \\
r = 3\ (d_{min} = 1): & s = 0 & \mu = (\mu_0,\ \mu_1,\ \mu_2,\ \mu_3,\ \mu_4,\ \mu_5,\ \mu_6,\ \mu_7) \quad (k = 8)
\end{array}$$

Example 4.10. *Consider P_{3^2} in Example 4.3 (hence, $n = 3^2 = 9$) and a message vector $\mu = (\mu_0,\ \mu_1,\ ...,\ \mu_8)$ and let s be the number of consecutive zeros in the vector μ for a given value of r:*

$$\begin{array}{lll}
r = 0\ (d_{min} = 9): & s = 8 & \mu = (\mu_0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0) \quad (k = 1) \\
r = 1\ (d_{min} = 6): & s = 5 & \mu = (\mu_0,\ \mu_1,\ 0,\ \mu_3,\ 0,\ 0,\ 0,\ 0,\ 0) \quad (k = 3) \\
r = 2\ (d_{min} = 3): & s = 2 & \mu = (\mu_0,\ \mu_1,\ \mu_2,\ \mu_3,\ \mu_4,\ 0,\ \mu_6,\ 0,\ 0) \quad (k = 6) \\
r = 3\ (d_{min} = 2): & s = 1 & \mu = (\mu_0,\ \mu_1,\ \mu_2,\ \mu_3,\ \mu_4,\ \mu_5,\ \mu_6,\ \mu_7,\ 0) \quad (k = 8) \\
r = 4\ (d_{min} = 1): & s = 0 & \mu = (\mu_0,\ \mu_1,\ \mu_2,\ \mu_3,\ \mu_4,\ \mu_5,\ \mu_6,\ \mu_7,\ \mu_8) \quad (k = 9)
\end{array}$$

In the above examples, d_{min} is shown in parentheses for each value of r; furthermore, observe that $d_{min} = s + 1$. Recalling for a moment the GFFT approach to Reed-Solomon code design, the minimum distance of a code where the message vector has $n-k$ consecutive zeros can be shown to be $d_{min} = n - k + 1$ (Blahut (2003); Wicker (1994)). It is apparent that, by using a Pascal matrix as the transform, a result similar to that of the GFFT can be ascertained. The major difference is that, for Reed-Solomon codes, the string of zeros must occur at the end of the message vector before applying the GFFT to create C. For $P(r,m)$, in addition to the string of consecutive zeros, based upon the structure of P_{p^m}, zeros must also be dispersed in other positions within μ to form code vectors $C = \mu P_{p^m}$.

5. Extensions of the Fourier convolution theorem over finite fields

The convolution operation involves relating the componentwise product of two vectors in one domain to the convolution of their transforms (Blahut & Burrus (1991)). Many linear transforms have well-defined convolution operations. For instance, the Hadamard transform yields the so-called logical or 'dyadic' convolution operation (Ahmed et al. (1973); Dodd (2003); Robinson (1972)). In this chapter, we develop extensions of the convolution theorem that can be used to reveal useful properties of other classes of codes. As an example, we demonstrate how the GFFT approach can be applied to describe generalized Reed-Muller codes (Blahut (2003)).

To begin the formulation, we consider the componentwise product $\gamma_j = \mu_j \lambda_j$ of two vectors $\mu = (\mu_0 \ ... \ \mu_{n-1})$ and $\lambda = (\lambda_0 \ ... \ \lambda_{n-1})$. Furthermore, we consider matrix transforms such that $C \equiv \mu P_{p^m}$ and $\Lambda \equiv \lambda P_{p^m}$ or, equivalently, $\mu = C Q_{p^m}$ and $\lambda = \Lambda Q_{p^m}$ where $(P_{p^m})^{-1} \equiv Q_{p^m}$. (here, '$\mu$' denotes the message vector and 'C' denotes the code vector). We demonstrate a formulation analogous to the convolution operation that describes $\gamma = \Gamma Q_{p^m}$:

$$\begin{aligned}
\Gamma_i &= \sum_{j=0}^{n-1} \gamma_j p_{ji} \mod p \qquad i = 0,\ 1,\ ...,\ n-1 \\
&= \sum_{j=0}^{n-1} (\mu_j \lambda_j) p_{ji} \mod p \\
&= \sum_{j=0}^{n-1} \mu_j \Big(\sum_{k=0}^{n-1} \Lambda_k q_{kj} \Big) p_{ji} \mod p \\
&= \sum_{k=0}^{n-1} \Lambda_k \Big(\sum_{j=0}^{n-1} \mu_j q_{kj} p_{ji} \Big) \mod p \\
&\equiv \sum_{k=0}^{n-1} \Lambda_k T_{i,k} \mod p \qquad i = 0,\ 1,\ ...,\ n-1
\end{aligned} \tag{17}$$

where $n = p^m$.

Notice that if we are dealing with familiar spectral transforms such as the Fourier or the Hadamard transform (where P denotes the forward transform and Q denotes the inverse transform), $T_{i,k}$ takes on a simple form. This is because the product $q_{kj} p_{ji}$ in $\sum_{j=0}^{n-1} \mu_j q_{kj} p_{ji}$ reduces to a term that enables us to take the transform of μ as $C_{f(i,k)} = \frac{1}{n} (\sum_{j=0}^{n-1} \mu_j p_{j,f(i,k)})$. For the case of the Fourier transform $f(i,k) = i - k$ and $T_{i,k} = C_{(i-k)}$; as expected, one ends up with the convolution theorem (Blahut (2003); Wicker (1994)). In the case of a Hadamard transform, $f(i,k) = i \oplus k$ (where $\oplus$ denotes bit-by-bit addition of the binary expansions of i and k) and $T_{i,k} = C_{(i \oplus k)}$. Here, the bit-by-bit addition $\oplus$ of the binary expansions of i and k over GF(2) would result in the dyadic convolution (Ahmed et al. (1973); Robinson (1972)).

For the codes in this presentation, the $q_{kj} p_{ji}$ term in the above summation leads to a convolution theorem that depends on the matrix P_{p^m}. Furthermore, this theorem can also be applied to demonstrate how to decode C to recover the message vector μ. In Equation (17)

$q_{kj} = (-1)^{j-k}\binom{j}{k} \mod p$ and $p_{ji} = \binom{i}{j} \mod p$; therefore, the product $q_{kj}p_{ji}$ will not lead to an expression that readily reduces the inner summation to a single term. To see why, let's write out $T_{i,k}$ as follows:

$$\begin{aligned} T_i &= (T_{i,0}\ T_{i,1}\ ...\ T_{i,n-1}) \\ &= (\mu_0\ \mu_1\ ...\ \mu_{n-1}) \begin{bmatrix} q_{00}p_{0i} & q_{10}p_{0i} & \cdots & q_{(n-1)0}p_{0i} \\ q_{01}p_{1i} & q_{11}p_{1i} & \cdots & q_{(n-1)1}p_{1i} \\ \vdots & \vdots & \cdots & \vdots \\ q_{0(n-1)}p_{(n-1)i} & q_{1(n-1)}p_{(n-1)i} & \cdots & q_{(n-1)(n-1)}p_{(n-1)i} \end{bmatrix} \\ &= (\mu_0\ \mu_1\ ...\ \mu_{n-1}) \begin{bmatrix} p_{0i} & & & \\ & p_{1i} & & \\ & & \ddots & \\ & & & p_{(n-1)i} \end{bmatrix} \begin{bmatrix} q_{00} & q_{10} & \cdots & q_{(n-1)0} \\ q_{01} & q_{11} & \cdots & q_{(n-1)1} \\ \vdots & \vdots & \cdots & \vdots \\ q_{0(n-1)} & q_{1(n-1)} & \cdots & q_{(n-1)(n-1)} \end{bmatrix} \\ &\equiv \mu D_i Q_{p^m}^T \end{aligned} \tag{18}$$

where T denotes the matrix transpose.

Observation 5.1. *The components of the vector $T_i = (T_{i,0}\ T_{i,1}\ ...\ T_{i,n-1})$ can be written as a linear combination of the components of $C = (C_0\ ...\ C_{n-1})$.*

Proof: Let

$$M_i \equiv D_i Q_{p^m}^T \tag{19}$$

where D_i is defined in Equation (18) and

$$\begin{aligned} A_i &\equiv Q_{p^m} M_i = Q_{p^m} D_i Q_{p^m}^T \\ &\Rightarrow M_i = P_{p^m} A_i. \end{aligned} \tag{20}$$

Then,

$$T_i = \mu M_i = \mu P_{p^m} A_i = C A_i. \tag{21}$$

Combining this result with Equation (17) we conclude

$$\begin{aligned} \Gamma_i &= \sum_{k=0}^{n-1} \Lambda_k T_{i,k} \mod p \qquad i = 0,\ 1,\ ...,\ n-1 \\ &= \sum_{k=0}^{n-1} \Lambda_k (CA_i)_k \mod p \qquad i = 0,\ 1,\ ...,\ n-1 \end{aligned} \tag{22}$$

So, instead of $T_{i,k}$ reducing to one single component of the vector C (as one might expect from a typical convolution operation), the Pascal convolution requires a linear combination of the components of C. Although this operation is slightly more complicated than the Fourier approach, the identity in Equation (8) does induce a simplification.

Observation 5.2. *(Symbolic Computation of Pascal Convolution)*
For the 1st order case where $n = p$ and $i = 0,\ ...,\ p-1$, using Equation (19) let $\hat{M}_i \equiv M_i$,

using Equation (18) let $\hat{D}_i \equiv D_i$ *and let* $\hat{A}_i \equiv Q_p \hat{M}_i$. *Then, for any* $0 \leq j \leq p^m - 1$ *where* $j = j_0 p^0 + j_1 p^1 + ... + j_{m-1} p^{m-1}$ *and* $A_j = Q_{p^m} M_j$,

$$A_j = \hat{A}_{j_{m-1}} \otimes ... \otimes \hat{A}_{j_1} \otimes \hat{A}_{j_0} \tag{23}$$

where $M_j \equiv D_j Q_{p^m}^T$.

Proof: The statement is clearly true for the first order case $m = 1$ since $j = j_0$. By induction let $j = j_0 p^0 + j_1 p^1 + ... + j_{m-1} p^{m-1}$ and assume that

$$D_j = \hat{D}_{j_{m-1}} \otimes ... \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0}$$

where $0 \leq j_k \leq p - 1$ for all $k = 0, ..., m - 1$. Consider any $j' = j_0 p^0 + ... + j_{m-1} p^{m-1} + j_m p^m$ and apply Equation (18) along with Lucas' theorem to obtain the following intermediate result:

$$\begin{aligned}
\hat{D}_{j_m} \otimes \hat{D}_{j_{m-1}} \otimes ... \otimes \hat{D}_{j_0} &= \hat{D}_{j_m} \otimes D_j \\
&= \begin{bmatrix} \binom{j_m}{0} & & & \\ & \binom{j_m}{1} & & \\ & & \ddots & \\ & & & \binom{j_m}{p-1} \end{bmatrix} \otimes \begin{bmatrix} \binom{j}{0} & & & \\ & \binom{j}{1} & & \\ & & \ddots & \\ & & & \binom{j}{p^m-1} \end{bmatrix} \\
&= \begin{bmatrix} \binom{j'}{0} & & & \\ & \binom{j'}{1} & & \\ & & \ddots & \\ & & & \binom{j'}{p^{m+1}-1} \end{bmatrix} \\
&= D_{j'}
\end{aligned} \tag{24}$$

Therefore, $D_j = \hat{D}_{j_{m-1}} \otimes ... \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0}$ is true. Next, successively apply the identity $(AC) \otimes (BD) = (A \otimes B)(C \otimes D)$ to obtain:

$$\begin{aligned}
\hat{M}_{j_{m-1}} \otimes ... \otimes \hat{M}_{j_1} \otimes \hat{M}_{j_0} &= (\hat{D}_{j_{m-1}} Q_p^T) \otimes ... \otimes (\hat{D}_{j_1} Q_p^T) \otimes (\hat{D}_{j_0} Q_p^T) \\
&= (\hat{D}_{j_{m-1}} \otimes ... \otimes \hat{D}_{j_1} \otimes \hat{D}_{j_0})(Q_p^T \otimes Q_p^T \otimes ... \otimes Q_p^T) \\
&= D_j Q_{p^m}^T \\
&= M_j
\end{aligned}$$

Finally, we arrive at the desired conclusion

$$\begin{aligned}
(\hat{A}_{j_{m-1}} \otimes ... \otimes \hat{A}_{j_1} \otimes \hat{A}_{j_0}) &= (Q_p \hat{M}_{j_{m-1}}) \otimes ... \otimes (Q_p \hat{M}_{j_1}) \otimes (Q_p \hat{M}_{j_0}) \\
&= (Q_p \otimes Q_p \otimes ... Q_p)(\hat{M}_{j_{m-1}} \otimes ... \otimes \hat{M}_{j_1} \otimes \hat{M}_{j_0}) \\
&= Q_{p^m} M_j \\
&= A_j.
\end{aligned}$$

Observation 5.2 tells us that, in order to calculate $T_j = CA_j$ for arbitrary $n = p^m$, one need only calculate $\hat{A}_i$ for $i = 0, \ ..., \ p-1$ and then take successive Kronecker products. The initial set of $\hat{A}_i$ for $i = 0, \ ..., \ p-1$ can easily be calculated by referring back to Equation (20) where $\hat{A}_i = Q_p \hat{M}_i = Q_p \hat{D}_i Q_p^T$.

An interesting property concerning the A_i is that the sum

$$\sum_{i=0}^{p^m-1} A_i = \sum_{i=0}^{p^m-1} Q_p \hat{D}_i Q_p^T$$

(where the sum is taken mod p) is a matrix of ones. This follows from two observations. First, from the definition of D_i in Equation (18), $\sum_{i=0}^{p^m-1} D_i$ is a matrix whose (p^m-1, p^m-1) entry is one and all other entries are zero. Second, it can also be demonstrated that the last column of Q_{p^m} must be a column of ones. Therefore, $Q_p \sum_{i=0}^{p^m-1} \hat{D}_i Q_p^T = \sum_{i=0}^{p^m-1} A_i$ is a matrix of ones.

Example 5.3. *For $p = 2$, the 1st order case $n = p$ gives $i = 0, 1$; hence, over $GF(2)$,*

$$P_p = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad Q_p = P_p = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix},$$

we calculate

$$\hat{A}_0 = Q_p \hat{D}_0 Q_p^T = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\hat{A}_1 = Q_p \hat{D}_1 Q_p^T = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

From Observation 5.2, to obtain the A_j for $n = p^2$ and $j = 0, 1, 2, 3$, one need only take successive Kronecker products as:

$$A_0 = \hat{A}_0 \otimes \hat{A}_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad A_1 = \hat{A}_0 \otimes \hat{A}_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_2 = \hat{A}_1 \otimes \hat{A}_0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad A_3 = \hat{A}_1 \otimes \hat{A}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

As expected, the A_i are symmetric matrices. Also, notice, as mentioned above, that $\sum_{i=0}^{p^m-1} A_i$ is a matrix of ones. For the case where $n = p^2$, let us now apply Observation 5.1 to calculate the Pascal convolution of the vectors $C = (C_0, C_1, C_2, C_3)$ and $\Lambda = (\Lambda_0, \Lambda_1, \Lambda_2, \Lambda_3)$. Using Equation (22), we have:

$$\begin{aligned} \Gamma_0 &= \Lambda_0 C_0 + \Lambda_1(0) + \Lambda_2(0) + \Lambda_3(0) \\ \Gamma_1 &= \Lambda_0 C_1 + \Lambda_1(C_0 + C_1) + \Lambda_2(0) + \Lambda_3(0) \\ \Gamma_2 &= \Lambda_0 C_2 + \Lambda_1(0) + \Lambda_2(C_0 + C_2) + \Lambda_3(0) \\ \Gamma_3 &= \Lambda_0 C_3 + \Lambda_1(C_2 + C_3) + \Lambda_2(C_1 + C_3) + \Lambda_3(C_0 + C_1 + C_2 + C_3). \end{aligned} \tag{25}$$

To close this section, we draw some immediate conclusions from Equation (25):

- Because of the Kronecker product, a good deal of self-similar structure can be observed in the resulting vector Γ. For instance, the coefficients of the Λ_i can be computed by iteration starting with the initial 'seed' generated by $\hat{A}_0$ and $\hat{A}_1$. As an example, the coefficient of Λ_1 in Γ_1 can be computed by adding the coefficient of Λ_0 in Γ_0 to the coefficient of Λ_0 in Γ_1. The coefficients of Λ_2 and Λ_3 in Γ_2 and Γ_3 can be computed by adding the coefficients of Λ_0 and Λ_1 in Γ_0 and Γ_1 to the coefficients of Λ_0 and Λ_1 in Γ_2 and Γ_3, and so on.
- Looking at the result columnwise, the set of coefficients associated with a given Λ_i appear to be the checksums for an $R(r, 2)$ binary Reed-Muller code ((MacWilliams & Sloane, 1977, p.385-388), (Wicker, 1994, p.155-165)). As pointed out in the next section, although this observation is true for the binary case, an orthogonal set of checksums for $p > 2$ will not come about by this method. It is the dual of the Pascal convolution that will lead to the decoding of GRM codes.

6. Majority logic decoding using Pascal convolution

GRM codes fall into a larger category of codes known as Euclidean geometry codes (Blahut (2003); Lin & Costello (1983); MacWilliams & Sloane (1977); Wicker (1994)) where it is well-known that a technique known as 'majority logic decoding' (MLD) can be used to recover the message vector. Based upon statements made in Section 4, it should be clear that Pascal codes are also MLD. However, the role played by the Pascal convolution in the decoding strategy is worthy of mention. As pointed out in the conclusions of Example 5.3, the checksums of a majority logic decoding (MLD) scheme for GRM codes can be derived using the dual of the convolution relation derived above. We now demonstrate this observation more clearly.

Because of the similar forms of P_{p^m} and Q_{p^m} the dual convolution relation is easily derived from the inverse transform. Consider the componentwise product $\Gamma_j = C_j \Lambda_j$ of two vectors where $C = \mu P_{p^m}$ and $\Lambda = \lambda P_{p^m}$:

$$\begin{aligned}
\gamma_i &= \sum_{j=0}^{n-1} \Gamma_j q_{ji} \mod p \qquad i = 0,\ 1,\ ...,\ n-1 \\
&= \sum_{j=0}^{n-1} (C_j \Lambda_j) q_{ji} \mod p \\
&= \sum_{j=0}^{n-1} C_j \Big(\sum_{k=0}^{n-1} \lambda_k p_{kj} \Big) q_{ji} \mod p \qquad (26) \\
&= \sum_{k=0}^{n-1} \lambda_k \Big(\sum_{j=0}^{n-1} C_j p_{kj} q_{ji} \Big) \mod p \\
&\equiv \sum_{k=0}^{n-1} \lambda_k s_{i,k} \mod p \qquad i = 0,\ 1,\ ...,\ n-1
\end{aligned}$$

where $n = p^m$. Similar to Equation (18), one can also show that

$$\begin{aligned} s_i &= (s_{i,0}\ s_{i,1}\ ...\ s_{i,n-1}) \\ &= C\Delta_i P_{p^m}^T \end{aligned} \tag{27}$$

which can also be written as

$$s_i = \mu P_{p^m} \Delta_i P_{p^m}^T \tag{28}$$

where Δ_i is a diagonal matrix with elements $(q_{0i}\ \ q_{1i}\ \ ...\ \ q_{(n-1)i})$ along its diagonal. Furthermore, if we define

$$B_i \equiv P_{p^m} \Delta_i P_{p^m}^T \tag{29}$$

then results similar to Observations 5.1 and 5.2 can also be demonstrated. However, in proving the dual of Observation 5.2 there is one difference be aware of. Since $q_{ji} = (-1)^{i-j}\binom{i}{j}$, the Kronecker product in the dual of Equation (24) will contain extra factors of $(-1)^{i-j}$. To achieve the equality $\Delta_j = \hat{\Delta}_{j_{m-1}} \otimes ... \otimes \hat{\Delta}_{j_1} \otimes \hat{\Delta}_{j_0}$ where $j = j_0 p^0 + j_1 p^1 + ... + j_{m-1} p^{m-1}$ the following identity will be required:

$$\begin{aligned} (-1)^k &= (-1)^{k_0 p^0 + k_1 p^1 + ... + k_{m-1} p^{m-1}} \\ &= (-1)^{k_0} ((-1)^p)^{k_1} ((-1)^{p^2})^{k_2} ... ((-1)^{p^{m-1}})^{k_{m-1}} \\ &= (-1)^{\sum_{l=0}^{m-1} k_l} \end{aligned}$$

for any $0 \le k \le p^m - 1$ where we have applied $a^p = a$ for any $a \in GF(p)$. Then, following the proof of Observation 5.2, it is straightforward to show that for any $0 \le j \le p^m - 1$ where $j = j_0 p^0 + j_1 p^1 + ... + j_{m-1} p^{m-1}$,

$$B_j = \hat{B}_{j_{m-1}} \otimes ... \otimes \hat{B}_{j_1} \otimes \hat{B}_{j_0} \tag{30}$$

where

$$\hat{B}_{j_k} = P_p \hat{\Delta}_{j_k} P_p^T.$$

In Section 4, we explained that the form of message vectors when applying P_{p^m} as the transformation where the message vector $\mu = (\mu_0,\ ...,\ \mu_{p^m-1})$ should have all components $\mu_j = 0$ if $w_p(j) > r$ (see Examples 4.9 and 4.10). To see how this formulation can lead to a decoding scheme, let us examine the case where $p = 2$, $m = 2$ and $r = 1$ (i.e. - a 1^{st} order binary Reed-Muller code of length 4). Consider first using Equations (26) and (27) to calculate Pascal convolution of the vectors $\mu = (\mu_0, \mu_1, \mu_2, \mu_3)$ and $\lambda = (\lambda_0, \lambda_1, \lambda_2, \lambda_3)$:

$$\begin{array}{lllllll} 00: & \gamma_0 = & \lambda_0 C_0 & + & \lambda_1(0) & + & \lambda_2(0) & + \lambda_3(0) \\ 01: & \gamma_1 = & \lambda_0(C_0 + C_1) & + & \lambda_1 C_1 & + & \lambda_2(0) & + \lambda_3(0) \\ 10: & \gamma_2 = & \lambda_0(C_0 + C_2) & + & \lambda_1(0) & + & \lambda_2 C_2 & + \lambda_3(0) \\ 11: & \gamma_3 = & \lambda_0(\sum_{i=0}^{3} C_i) & + & \lambda_1(C_1 + C_3) & + & \lambda_2(C_2 + C_3) & + \lambda_3 C_3 \end{array} \tag{31}$$

where the binary expansion of the γ index has been explicitly written out at the beginning of each row. Next, consider Equations (26) and (28) to calculate the same convolution:

$$\begin{array}{ll} 00: & \gamma_0 = \lambda_0\mu_0 + \quad \lambda_1(0) \quad + \quad \lambda_2(0) \quad + \quad \lambda_3(0) \\ 01: & \gamma_1 = \lambda_0\mu_1 + \lambda_1(\mu_0+\mu_1) + \quad \lambda_2(0) \quad + \quad \lambda_3(0) \\ 10: & \gamma_2 = \lambda_0\mu_2 + \quad \lambda_1(0) \quad + \lambda_2(\mu_0+\mu_2) + \quad \lambda_3(0) \\ 11: & \gamma_3 = \lambda_0\mu_3 + \lambda_1(\mu_2+\mu_3) + \lambda_2(\mu_1+\mu_3) + \lambda_3(\sum_{i=0}^{3}\mu_i) \end{array} .$$

Since, for $P_2(1,2)$, $\mu = (\mu_0,\ \mu_1,\ \mu_2,\ 0)$, this set of equations can be simplified as

$$\begin{array}{ll} 00: & \gamma_0 = \lambda_0\mu_0 + \quad \lambda_1(0) \quad + \quad \lambda_2(0) \quad + \quad \lambda_3(0) \\ 01: & \gamma_1 = \boldsymbol{\lambda_0\mu_1} + \lambda_1(\mu_0+\mu_1) + \quad \lambda_2(0) \quad + \quad \lambda_3(0) \\ 10: & \gamma_2 = \lambda_0\mu_2 + \quad \boldsymbol{\lambda_1(0)} \quad + \lambda_2(\mu_0+\mu_2) + \quad \lambda_3(0) \\ 11: & \gamma_3 = \lambda_0(0) + \quad \lambda_1\mu_2 \quad + \quad \boldsymbol{\lambda_2\mu_1} \quad + \lambda_3(\mu_0+\mu_1+\mu_2) \end{array} . \tag{32}$$

Equations (31) and (32) must hold for *any* vector λ. Therefore, for a specific γ_j, we can equate the coefficients of the λ_i in Equation (31) with those in Equation (32). So, for example, we end with the result that

$$\mu_2 = C_0 + C_2$$
$$\mu_2 = C_1 + C_3$$

and

$$\mu_1 = C_0 + C_1$$
$$\mu_1 = C_2 + C_3.$$

For this first order $r = 1$ code, we can generate a set of checksums using a simple algorithm. Start at an index i of γ such that $w_2(i) = 1$ and equate Equations (31) and (32) along a *diagonal path* in order to 'collect' all checksum equations associated associated with μ_i. For example, the bold symbols in Equation (32) generate the checksums for μ_1. It turns out that these diagonal equations actually generate what are known as the 'incidence vectors' of the MLD strategy (Blahut (2003); MacWilliams & Sloane (1977); Wicker (1994)).

We now provide an algorithm for $GF(p)$ to show how the Pascal convolution approach is equivalent to a typical MLD using finite Euclidean geometry ((Wicker, 1994, p.155-165)). The interesting aspect of this algorithm is that the Pascal convolution generates the correct checksums for *any* $GF(p)$. Consider a $P_p(r,m)$ code where $C = \mu P_{p^m}$ such that $\mu_j = 0$ if $w_p(j) > r$:

(0) Let $j = r$.

(1) Let S_j be the set of indices i such that $w_p(i) = j$.

(2) Apply Equation (27) to calculate γ.

(3) Apply Equation (28) to calculate γ (these equations will simplify based upon which of the μ_i are zero).

(4) For each $i \in S_j$, start at λ_0 associated with γ_i and construct checksum equations by equating the result in Step (2) with that of Step (3) along a *diagonal path* (i.e. - starting at k=0, choose the coefficient of λ_k associated with γ_{i+k}).

(5) For $i \in S_j$, create estimates $\bar{\mu}_i$ by a majority logic decision on the checksums.

(6) $j = j - 1$. If $j < 0$, stop.

(7) Remove the estimated components as:

$$\bar{C} = \bar{\mu} P_{p^m}$$
$$\hat{C} \equiv C - \bar{C} \ (= (\mu - \bar{\mu}) P_{p^m}).$$

(8) Adjust μ to reflect the change in step (7) as follows. Construct a new vector $\tilde{\mu}$ where $\tilde{\mu}_i = \mu_i$ if $i \in S_j$ and $\tilde{\mu}_i = 0$ otherwise. Then let

$$\hat{\mu} \equiv \mu - \tilde{\mu}.$$

(9) Let $C = \hat{C}$ and $\mu = \hat{\mu}$ and go to Step (1).

As with typical MLD schemes, this algorithm starts with the highest order r to obtain estimates of the code vector components and then successively estimates the lower order components.

Example 6.1. *Let $p = 3$, $m = 2$ and $r = 2$. Consider decoding a $P_3(2,2)$ code. From Example 4.10,*

$$\mu = (\mu_0,\ \mu_1,\ \mu_2,\ \mu_3,\ \mu_4,\ 0,\ \mu_6,\ 0,\ 0).$$

Also, we know that $P_3(2,2)$ has $d_{min} = 3$ implying that we can correct a single error. Therefore, we expect that the MLD equations should have at least three checksums.

(0) Start with $j = 2$.

(1) Let $S_2 = \{2,4,6\}$ (i.e. - $i = i_0 + i_1 p$ such that $w_3(i) = 2$).

(2,3,4) Rather than write out the equations for γ_i, we summarize by equating the results of step (2) and step (3):

$i = 2$:

$$\mu_2 = c_0 + c_1 + c_2$$
$$\mu_2 = c_3 + c_4 + c_5$$
$$\mu_2 = c_6 + c_7 + c_8$$

$i = 4$:

$$\mu_4 = c_0 + 2c_1 + 2c_3 + c_4$$
$$2\mu_4 = 2c_1 + c_2 + c_4 + 2c_5$$
$$2\mu_4 = 2c_3 + c_4 + c_6 + 2c_7$$
$$\mu_4 = c_4 + 2c_5 + 2c_7 + c_8$$

$i = 6$:

$$\mu_6 = c_0 + c_3 + c_6$$
$$\mu_6 = c_1 + c_4 + c_7$$
$$\mu_6 = c_2 + c_5 + c_8$$

After estimating the message components dictated by S_2 (step (5)), remove the code estimates from C (step (7)) and begin work on S_1 where now (step(8)) $\mu_i = 0$ if $w_p(i) > 1$. For S_1, we have the checksums:

$$
\begin{aligned}
i = 1: \quad \mu_1 &= 2c_0 + c_1 \\
2\mu_1 &= c_1 + 2c_2 \\
\mu_1 &= 2c_3 + c_4 \\
2\mu_1 &= c_4 + 2c_5 \\
\mu_1 &= 2c_6 + c_7 \\
2\mu_1 &= c_7 + 2c_8
\end{aligned}
$$

$$
\begin{aligned}
i = 3: \quad \mu_3 &= 2c_0 + c_3 \\
\mu_3 &= 2c_1 + c_4 \\
\mu_3 &= 2c_2 + c_5 \\
2\mu_3 &= c_3 + 2c_6 \\
2\mu_3 &= c_4 + 2c_7 \\
2\mu_3 &= c_5 + 2c_8
\end{aligned}
$$

After estimating the message components dictated by S_1, once again, remove the code estimates from C and begin work on S_0 where now $\mu_i = 0$ if $w_p(i) > 0$. At this stage, with all other components of $\mu = 0$ except μ_0, we are left with $\mu = C$ (i.e. - nine estimate of the check on μ_0).

7. Conclusions

When considering the design of error control codes, it is interesting to look for guiding principles that can account for whole classes of codes. In this presentation, we have shown how the GFFT convolution approach to Reed-Solomon codes can be extended to other classes of codes such as generalized Reed-Muller codes.

Code	Convolution Principle	Decoding Strategy
Reed-Solomon	GFFT-based	iterative
GRM	generalized	iterative

Table 1. Comparison of Fourier and generalized convolution techniques.

Instead of applying a Fourier matrix to encode the message, we have applied a Pascal matrix and extended the convolution theorem over finite fields. In doing so, we have observed that this formulation leads to the well-known majority logic decoding algorithm. Additional investigations have also considered codes in the context of the wavelet transform (Sakk & Wicker (2003)). The block codes addressed in this chapter have been shown to lend themselves to graph-based iterative decoding strategies (see Table 1). The results derived above suggest that the generalized convolution approach is useful for understanding the systematic introduction of redundancy for the sake of error control.

8. References

Ahmed, N., Rao, K. & Abdussattar, A. (1973). On cyclic autocorrelation and the Walsh-Hadamard transform, *IEEE Transactions on Electromagnetic Compatibility* 18: 141–146.

Blahut, R. (2003). *Algebraic Codes for Data Transmission*, Cambridge University Press.

Blahut, R. & Burrus, C. (1991). *Algebraic Methods for Signal Processing and Communications Coding*, Springer.

Burrus, C., Gopinath, R. & Guo, H. (1998). *Introduction to Wavelets and Wavelet Transforms*, Prentice-Hall, NJ.

Caire, G., Grossman, R. & Poor, H. (1993). Wavelet transforms and associated finite cyclic groups, *IEEE Transactions on Information Theory* 39: 1157–1166.

Call, G. & Velleman, D. (1993). Pascal's matrices, *American Mathematical Monthly* 100: 372–376.

Dodd, M. (2003). *Applications of the Discrete Fourier Transform in Information Theory and Cryptology*, PhD thesis, Royal Holloway and Bedford New College, University of London.

Forney, G. D. (1988). Coset codes - Part II: Binary lattices and related codes, *IEEE Transactions on Information Theory* 34: 1152–1187.

Heller, S. (1963). Inverse of triangular matrix, *American Mathematical Monthly* 70: 334.

Kou, Y., Lin, S. & Fossorier, M. P. C. (2001). Low-density parity-check codes based on finite geometries: A rediscovery and new results, *IEEE Transactions on Information Theory* 47: 2711–2736.

Li, G., Li, D., Wang, Y. & Sun, W. (2010). Hybrid decoding of finite geometry low-density parity-check codes, *IET Communications* 4(10): 1238–1246.

Lin, S. & Costello, D. (1983). *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, New York.

Liu, Z. & Pados, D. A. (2005). Decoding algorithm for finite-geometry LDPC codes, *IEEE Transactions on Communications* 53: 415–421.

MacWilliams, F. J. & Sloane, N. J. A. (1977). *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam.

Massey, J. L., Costello, D. J. & Justesen, J. (1973). Polynomial weights and code constructions, *IEEE Transactions on Information Theory* 19: 101–110.

Ngatched, T. M. N., F, T. & Bossert, M. (2009). An improved decoding algorithm for finite-geometry LDPC codes, *IEEE Transactions on Communications* 57: 302–306.

O.Vontobel, P., Smarandache, R., Kiyavash, N., Teutsch, J. & D.Vukobratovic (2005). On the minimal pseudocodewords of codes from finite geometries, *Proc. IEEE Int. Symp. Inf. Theory*, Adelaide, Australia.

Pusane, A. E., Smarandache, R., Vontobel, P. O. & Costello, D. J. (2011). Deriving good LDPC convolutional codes from LDPC block codes, *IEEE Transactions on Information Theory* 57: 835–857.

Robinson, G. (1972). Logical convolution and discrete Walsh and Fourier power spectra, *IEEE Transactions on Audio and Electroacoustics* 20: 271–280.

Sakk, E. (2002). *Wavelet Packet Formulation of Generalized Reed Muller Codes*, PhD thesis, Cornell University, Ithaca, NY.

Sakk, E. & Wicker, S. (2003). Wavelet packets for error control coding, *Proceedings of the SPIE Volume 5207 (Wavelets X)*, San Diego, CA.

Smarandache, R., Pusane, A. E., Vontobel, P. O. & Costello, D. J. (2009). Pseudocodeword performance analysis for LDPC convolutional codes, *IEEE Transactions on Information Theory* 55: 2577–2598.

Strang, G. & Nuygen, T. (1996). *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, MA.

Tang, H., Xu, J., Lin, S. & Abdel-Ghaffar, K. A. S. (2005). Codes on finite geometries, *IEEE Transactions on Information Theory* 51: 572–596.

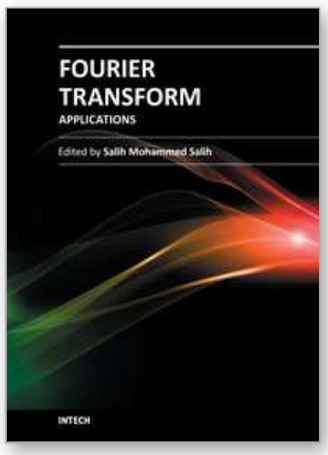
Vandendriesscher, P. (2010). Some low-density parity-check codes derived from finite geometries, *Designs, Codes and Cryptography* 54: 287–297.

Wicker, S. (1994). *Error Control Systems for Digital Communication and Storage*, Prentice Hall.

Wicker, S. & Kim, S. (2003). *Fundamentals of codes, graphs, and iterative decoding*, Kluwer.

Xia, S.-T. & Fu, F.-W. (2008). Minimum pseudoweight and minimum pseudocodewords of LDPC codes, *IEEE Transactions on Information Theory* 54: 480–485.

Zhang, L., Huang, Q. & Lin, S. (2010). Iterative decoding of a class of cyclic codess, *Information Theory and Applications Workshop (ITA)*, San Diego, CA.



Fourier Transform Applications

Edited by Dr Salih Salih

ISBN 978-953-51-0518-3
Hard cover, 300 pages
Publisher InTech
Published online 25, April, 2012
Published in print edition April, 2012

The book focuses on Fourier transform applications in electromagnetic field and microwave, medical applications, error control coding, methods for option pricing, and Helbert transform application. It is hoped that this book will provide the background, reference and incentive to encourage further research and results in these fields as well as provide tools for practical applications. It provides an applications-oriented analysis written primarily for electrical engineers, control engineers, signal processing engineers, medical researchers, and the academic researchers. In addition the graduate students will also find it useful as a reference for their research activities.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Eric Sakk and Schinnel Small (2012). The Fourier Convolution Theorem over Finite Fields: Extensions of its Application to Error Control Coding, Fourier Transform Applications, Dr Salih Salih (Ed.), ISBN: 978-953-51-0518-3, InTech, Available from: http://www.intechopen.com/books/fourier-transform-applications/the-fourier-convolution-theorem-over-finite-fields-extensions-of-its-application-to-error-control-c

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen