VOL. 15, NO. 12, JUNE 2020 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

PERFORMANCE ANALYSIS OF GEOMETRIC CODES

Nasser Abdellatif and Maryam Akho-Zahieh
Department of Electrical Engineering, Faculty of Engineering, Applied Science Private University, Amman, Jordan
E-Mail: nasser_abdellatif@asu.edu.jo

ABSTRACT

Forward error correcting codes have proven their usefulness over traditional communication channels and are of great benefit when conditions are normal. However, these codes perform very poorly over noisy or fading channels. Geometric Codes are classes of codes that outperform error-correcting codes over noisy channels.

Keywords: geometric codes, fading channels, code rate, parity symbol.

1. INTRODUCTION

Most error correcting codes perform well on channels with low error probability, but their efficiency drops as the channel becomes noisy (fades). Many techniques have been presented to fix this problem, including using adjustable rate codes. In this paper, we introduce a group of codes called the Geometric Codes, and show that though geometric codes are not optimal, they outperform many codes, particularly over noisy or fading channels (input error rates of 10^{-2} to 10^{-3}). We have tested our codes on the systems presented in [3][4][5][6][7] [8] [9]and [10] and have gotten significant improvement.

There are three classes of Geometric Codes: Basic Geometric Code (BGC), Partially Modified Geometric Codes (PGC), and Totally Modified Geometric Codes (TGC). The encoding and decoding algorithms, as well as possible hardware implementations for BGC and PGC, were presented in [1] and [2], respectively. We studied the performance of the TGC over a noisy channel. We then presented the simulation results for this code and have shown that the code performance does improve with the increase of the number of bits per symbol and with the coding rate and that it outperforms the other two classes of Geometric codes.

2. GEOMETRIC CODES

As shown in [1] The coding efficiency of these codes is k/k+r. We will describe the binary codes and later extend it for the nonbinary case.

In the Binary Basic Geometric Code, the location of each code bit is identified with a vertex of a uniformed lattice in the plane. Slopes of the form $1/m_1$, $1/m_2$, $1/m_3$, …, $1/m_r$ are chosen. To determine the jth parity row, a line having the slope $1/m_j$ is drawn through each data bit. The lines wrap cyclically upon reaching the left end of the row as though the block formed a cylinder. These lines are extended into the parity block, and each terminates at a particular parity bit in the jth parity row. This bit is determined so that the sum (mod 2) of the data bits on the line plus the single parity bit is zero. Thus the jth parity row is not affected by the other parity rows.

It follows from the construction that each data bit has r orthogonal estimators, corresponding to the r lines passing through it. Thus, if r is even, the code can correct r/2 errors by majority logic, while if r is odd (r-1)/2 errors may be corrected [1]

We impose a constraint on the slopes: no triangle with vertices at the bit positions and sides with slopes equal to $1/m_1$, $1/m_2$, $1/m_3$ should be constructible.

BGC provides no protection to the parity bits. The PGC partially solves this problem as follows: The first row of parity is determined as in BGC. The second parity row is found similarly, but the encoded first row of the parity is included in the parity equation. The third parity row checks both the first and second rows already determined. Continuing in this manner, each parity row checks that data rows, as well as the parity rows that lie above it.

This new scheme also provides r orthogonal estimator for each data bit and thus has a minimum distance of at least r+1. The minimum distance can be significantly improved, by correctly choosing the slopes $1/m_1$, $1/m_2$, …, $1/m_r$, the minimum distance of the PGC may be shown to be $2^r - 1$. This is accomplished by exhibiting $2^r - 2$ orthogonal equations for each bit. The procedure is illustrated in Figure 1 below. The r equations are given by considering the original (primary) parity equations. To obtain a new orthogonal equation, a substitution for each estimating bit is made in one of the primary equations, using an orthogonal estimator having different slope from the primary line slope [2].

This new line will be orthogonal to the primary equation if the substituted line does not intersect any primary line or any previously determined secondary lines. We impose a constraint on the slopes: no triangle with vertices at the bit positions and sides with slopes equal to $1/m_1$, $1/m_2$, $1/m_3$ should be constructible.

The third coding scheme, which is called TGC, requires that the parity bits be determined so that all bits along a line with slope $1/m_j$ have a zero sum. This condition cannot be produced directly as in the previous two codes, but the encoding problem may be approached algebraically as follows: Let us associate with each information row a polynomial $d_p(x)$, $(p = 1\ to\ k)$, and with each parity row a polynomial $c_q(x)$, $(q = 1\ to\ k)$.

It can be shown that the parity condition stated above leads to the equation:

$$\begin{pmatrix} 1 & x^{m_1} & x^{2m_1} & \cdots & x^{(k-1)m_1} \\ 1 & x^{m_2} & x^{2m_2} & \cdots & x^{(k-1)m_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x^{m_r} & x^{2m_r} & \cdots & x^{(k-1)m_r} \end{pmatrix} \begin{pmatrix} d_1(x) \\ d_2(x) \\ \vdots \\ d_k(x) \end{pmatrix} = \begin{pmatrix} x^{-m_1} & x^{-2m_1} & x^{2m_1} & \cdots & x^{-rm_1} \\ x^{-m_2} & x^{-2m_2} & x^{2m_2} & \cdots & x^{-rm_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x^{-m_r} & x^{-2m_r} & x^{2m_r} & \cdots & x^{-rm_r} \end{pmatrix} \begin{pmatrix} c_1(x) \\ c_2(x) \\ \vdots \\ c_r(x) \end{pmatrix} \quad (1)$$

The above equations are in the ring of polynomials modulo x^w+1. Because this is not a field, matrix inversion is not possible; however, the structure of the matrix does permit a solution, which is expressible as:

$$c_i(x) = \sum_{i=1}^{k} d_i(x) q_{ij}(x) \quad mod\ x^w + 1$$

where the $\{q_{ij}(x)\}$ are polynomial that depend upon the slopes. The above dertermination of the parity polynomial shows that the jth parity bit vector is the sum (mod 2) of the outputs of the k tapped cyclic shift registers. Implementation is, therefore, straightworward [1].

Each of the codes BGC, PGC, TGC described above as a block code have a convolutional code equivalent. Since, in the case of convolutional codes, there is no block length to consider, the slopes will never reach the end of the data, and therefore, wrap-around need not be used. Since wrapping is not used, some lines near the end of the data would require data bits that are non-existent. Whenever such a non-existent data bit is needed, it is assumed to be a zero bit. These 0-bits are not transmitted and, therefore, do not affect the code rate. This augmentation by zeros causes the parity lines to be longer than the data lines, thus, decrease the code rate. But for cases when the number of columns w is long so that $w \gg k$ and $w \gg r$, where k is the number of data rows and r is the number of parity rows, the decrease in the codes rate is negligible, and the code rate can be approximated by $R = \frac{k}{k+r}$ [2].

The decoding scheme for the TGC requires that each received bit has r syndrome bits associated with it, corresponding to the r primary orthogonal parity equations. The decoding algorithm begins by determining the number of the syndrome in the range $[0, r]$ is associated with each code bit position. The maximum of all such scores is an integer, n. If $n > 1$, all entries with score n are inverted. The process continues until $n \leq 1$. If the process terminates with $n = 1$, a decoding fault is indicated.

The TGC code is the most powerful of the three codes. The TGC code offers equal protection to the data lines and the parity lines. Hence, it allows the correction of errors occurring in the parity lines.

The simulation of the TGC block code is similar to the BGC and PGC described in [1] and [2], respectively. When a data block is received, it is augmented by zeros as needed. Except for the TGC code, the length of the all-zero blocks, λ, is longer than the length of the all-zero blocks needed for BGC and PGC codes. For the TGC code, the length of the all-zero blocks should be, $\lambda = (k + r - 1)m_r$. When this augmentation with zeros is done, no wrap-around is needed. When wrap-around is used the performance of the code will decrease since errors accuring around the edges may form uncorrectable error patterns which the augmentation by zeros avoids. Of course, the augmentation by zeros decreases the code rate, but when h, the number of columns in the block is too large, the change in code rate can be neglected.

In the encoder, since every parity line in the TGC code is dependent on all other parity lines, we would have to encode the data block in an exact manner. One way of accomplishing that is by the use of equation (1). Another way to encode that TGC code as described in Figure-2.

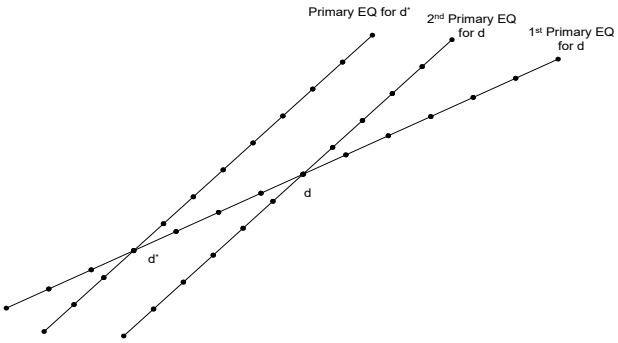

Figure-1. Construction of secondary equations (r=2).

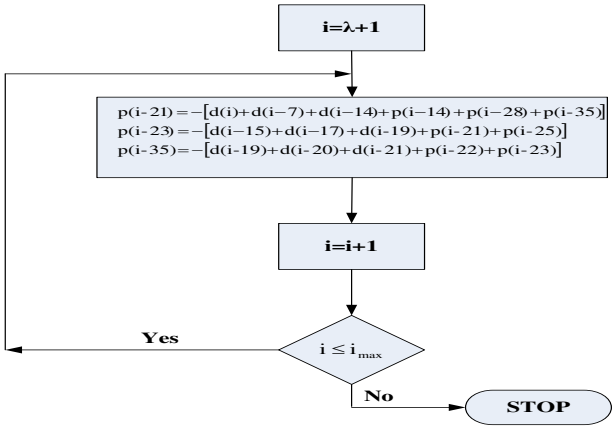

Figure-2. Flowchart

Here we start with the first symbol in the first data line and calculate the corresponding parity symbol in the first parity lines. The encoding equation for our particular example will be as follows:

$$p_1(15) = -[d_1(36) + d_2(29) + d_3(22) + p_2(8) + p_3(1)]$$

Where $p_2(8)$ and $p_3(1)$ are assumed to be zeros. To avoid finding which parity symbols need to be set to zero, we assume that every parity symbol of index $i < \lambda$ to be zero at the start. Usually, the number of symbols that need to be set to zero is small. Some of the values that we set to zero at the beginning need not be set zero, but their values to be calculated later are not affected by the initialization to zero. The second parity symbil calculated will be the symbol in the second parity line along the line of slop ½ that passes throught the symbol in the first parity line that was calculated before it. The encoding equations to find the second and third parity symbol will be

$$p_2(17) = -[d_1(9) + d_2(11) + d_3(13) + p_1(15) + p_3(19)]$$
$$p_3(18) = -[d_1(13) + d_2(14) + d_3(15) + p_1(16) + p_2(17)]$$

Note that we started this procedure with a line slope 1/7 beginning at the first data line, $d_1(\lambda + 1)$. The same procedure is then repeated until all data symbols are encoded. To encode the i^{th} symbol in the first data line, the equations, as shown in figure 2, will be:

$$p_1(i-21) = -[d_1(i) + d_2(i-7) + d_3(i-14) + p_2(i-28) + p_3(i-35)]$$
$$p_2(i-23) = -[d_1(i-15) + d_2(i-17) + d_3(i-19) + p_1(i-21) + p_3(i-25)]$$
$$p_3(i-24) = -[d_1(i-19) + d_2(i-20) + d_3(i-21) + p_1(i-22) + p_2(i-23)]$$

The decoder for the TGC block code is almost identical to those of the BGC and PGC codes. The only difference is in the decoding equations. The decoding equations for the ith symbol in the first data line in a TGC code are:

$$E_1 = -[d_1(i) + d_2(i-7) + d_3(i-14) + p_1(i-21) + p_2(i-28) + p_3(i-35)]$$
$$E_2 = -[d_1(i-15) + d_2(i-17) + d_3(i-19) + p_1(i-21) + p_2(i-23) + p_3(i-25)]$$
$$E_3 = -[d_1(i-19) + d_2(i-20) + d_3(i-21) + p_1(i-22) + p_2(i-23) + p_3(i-24)]$$

3. SIMULATION RESULTS

As for the cases of the BGC and PGC codes, we simulated many TGC codes. The behavior of these codes with respect to the changing parameters r, the number of parity lines, and m, the number of bits per symbol is shown in figures 3, 4 and 5 to be similar to the behavior of the BGC and PGC codes [1][2].

Figure-6 shows the m-ary TGC codes with different rates, all using 3-parity lines. Note that as expected, the lower rate codes outperform the higher rate codes. Also, note that the rate 9/10 code starts to show a coding gain around input probability of error of 10^{-3} and that its curve is descending at a slope comparable to those of the rate 1/2 and 3/4 codes.

Earlier, we argued that the TGC code is the most powerful of the three codes because it offers the most protection to parity symbols. Whereas the PGC code provides partial protection, and the BGC code provides no protection at all. Of course, all three codes give the same protection to the data symbols. Simulation results support our argument. Figures 7 and 8 compare the performance of the three coding schemes. Figure-7 compares the rate 1/2, while figure 8 compares the rate 3/4 binary BGC, PGC, and TGC codes. The figures show that the TGC outperforms PGC, which in turn outperforms the BGC. Figure-9 compares the three coding schemes to the Reed-Solomon code. In this case, a block of 155 bits was used for the RS code, where block sizes of 168, 192, 192, were used for the BGC, PGC, and TGC, respectively. The figure shows that Geometric Codes specially TGC and PGC start to outperform the RS code at an input error rate of about 10^{-2}. Also, Geometric codes have steeper slopes and hence, would have larger coding gains as the input error rate decreases beyond 10^{-2}.

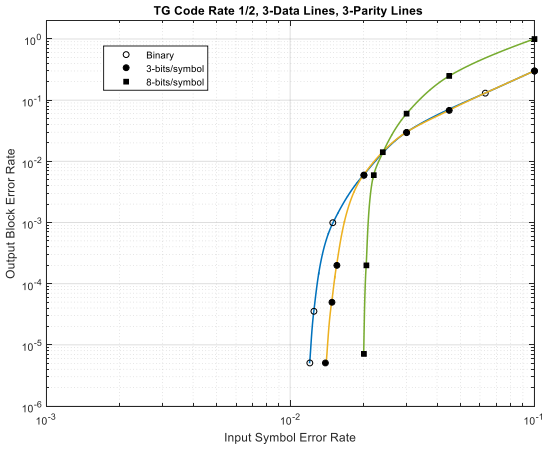

Figure-3.

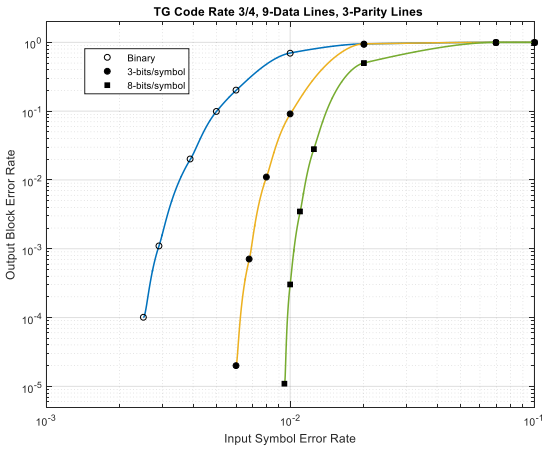

Figure-4.

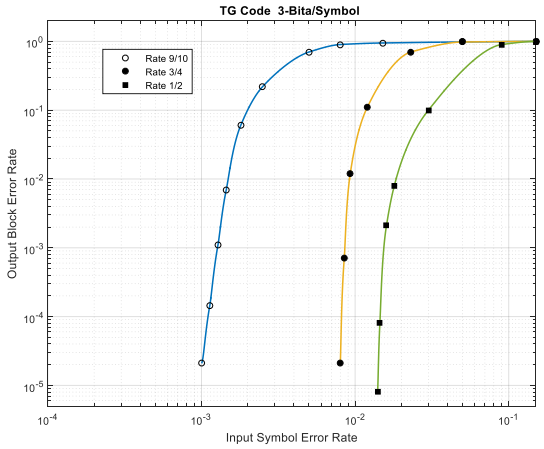

Figure-5.

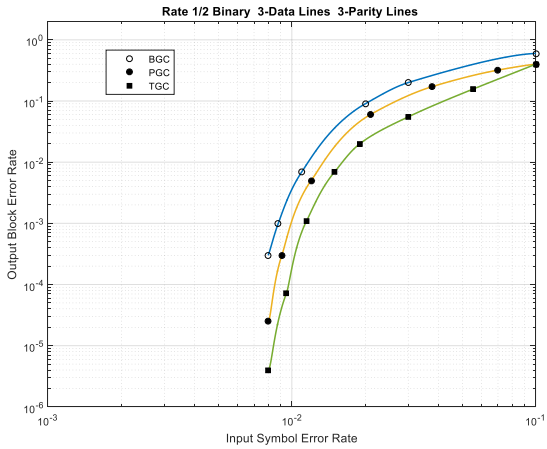

Figure-6.

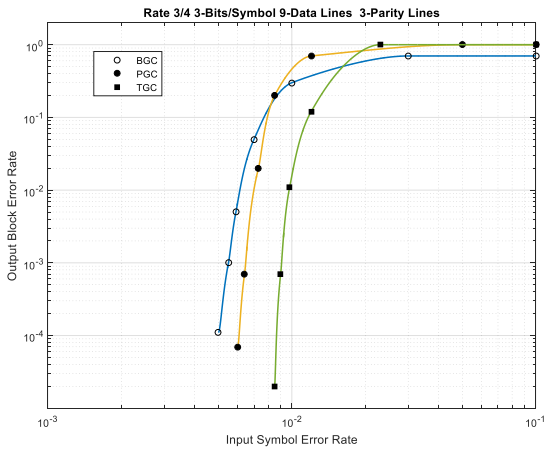

Figure-7.

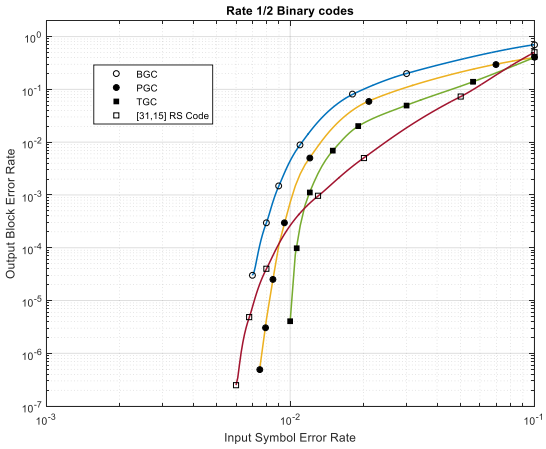

Figure-8.

4. CONCLUSIONS

This paper has been directed towards the study of the performance of Geometric Codes. We show that though geometric codes are not optimal, they outperform many codes, particularly over noisy or fading channels (input error rates of 10^{-2} to 10^{-3}). There are three types of Geometric codes BGC, PGC, and TGC. The three codes differ from each other by the amount of protection they offer for the parity symbols. The coding and decoding procedures of all three classes of codes were presented, and some theoretical description of the TGC was shown.

The effects of various code parameters are presented, which show that the output improves with the code rate, the number of bits/symbol, and the number of parity lines. Also, we compare the performance of the three codes and show that the TGC outperforms the BGC and PGC, which is expected since the TGC offers the most protection to the parity symbols than the others.

These codes offer simple coding and decoding implementations. Because of their flexibility, geometric codes can be tailored for use in many communication systems.

ACKNOWLEDGMENT

The authors are grateful to the Applied Science Private University, Amman, Jordan, for the full financial support granted to this research project.

REFERENCES

[1] Osama Nashwan, Sammer Issa and Khalil Saleh. 2019. Performance of Basic Geometric Codes over Fading Channels. ARPN Journal of Engineering and Applied Sciences. 14(19).

[2] Samer Issa and Osama Nashwan. September 2019.Performance Analysis of Modified Geometric Codes. ARPN Jornal of Engineering and Applied Sciences. 14(18).

[3] Maryam Akho-Zahieh and Nasser Abdellatif. 2014. Performance analysis of wavelet packet multirate multicarrier multicode CDMA wireless multimedia communications. International Journal of Electronics and Communications. 68(4): 268-277.

[4] Maryam Akho-Zahieh and Nasser Abdellatif. 2012. Narrow-band interference suppression in wavelet packets based multicarrier multicode CDMA overlay system. Wireless Communications, and Mobile Computing. DOI: 10.1002/wcm.2325.

[5] Maryam Akho-Zahieh and Nasser Abdellatif. 2013. Effect of jamming and diversity on the performance of a wavelet packet based multicarrier multicode CDMA communication system. International Journal of Electronics and Communications. 67(2): 157-166.

[6] Maryam Akho-Zahieh, Khalil Saleh and Nasser Abdellatif. 2013. Outage Probability Performance of Wavelet Packet Multirate Multicarrier Multicode CDMA System. The Jordan Journal of Applied Science. 11(1).

[7] Nasser Abdellatif and Maryam Akho-Zahieh. 2013. Effect of Narrowband Interference on the Performance of Wavelet Packets Based Multicarrier CDMA System With Decorrelating Multiuser Detector. The Jordan Journal of Applied Science. 11(1).

[8] Maryam Akho-Zahieh, Khalil Saleh and Nasser Abdellatif. Performance of Wavelet Packets Based Multicarrier Multicode CDMA Communication System in the Presence of Narrowband Jamming, 1st International Conference on Communications, Signal Processing and their Applications, Sharjah, UAE, 2013 1st International Conference on Communications, Signal Processing and their Applications, Sharjah, UAE, Digital Object Identifier: 10.1109/ICCSPA.2013.6487223, IEEE Conference Publications, 2013.

[9] Maryam Akho-Zahieh and Nasser Abdellatif. 2015. Effect of Diversity and Filtering on the Performance of Wavelet Packets Based Multicarrier Multicode CDMA System. Journal of Signal and Information Processing. 6, 165-179.