Chapter 1

Introduction

1.1 The evolution of coding theory

With the increasing usage of technological devices in our daily lives, it is crucial to get signals right. Many disciplines are collaborating to make this process go smoothly; electronic engineering, mathematics and computing are the foundations of this mechanism. The invention of the telegraph and telephone drew the attention of information scientists all over the world to the field's potential. The telegraph (the transmission of electrical signals through wires) was invented in the early 1700s, while many scientists claimed to have worked on the signals, it was New York University professor Samuel Finley Breese Morse who made the crucial breakthrough. Samuel Morse built a telegraph line between Washington, D.C. and Baltimore, Maryland in 1844, but he noticed a lot of issues with the electrical signals that were sent through the underground wires. Incomprehensibly, putting the wires on poles resulted in fewer complications. Similarly, in 1875, Alexander Graham Bell invented the telephone, which again drew the attention of numerous scientists looking for a way to solve the problem of electrical signals carried across the wires.

Information and coding theory are two branches of mathematics in which the transmission of information is studied. The formal study of information theory began with the papers "Certain Factors Affecting Telegraph Speed" [67] and "Transmission of Information" [41], written by Nyquist and Hartley respectively, in 1924 and 1928 at the Bell Laboratories in the United States of America. Coding theory is the subject dealing with error detection and correction for data that efficiently and accurately passes through a noisy communication channel. While working at Bell Laboratories in the United States of America, Claude E. Shannon, often recognized as the father of modern digital communications and information theory, produced the seminal work "A Mathematical Theory of Communication" [98] in 1948. This study is credited with bringing Information Theory to its initial evolutionary stage.

In a communication channel, information sent over it may get corrupted due to various noise factors. Shannon discovered a number, called the channel capacity and demonstrated that at any rate below the channel capacity, reliable communication is achievable, whereas, reliable communication is not possible if the transmission rate is greater than the channel capacity. Shannon's results guarantee that data may be encoded before transmission and then decoded to a certain degree of accuracy at the receiver's end. According to Shannon, in every communication channel, signals can be encoded before being transmitted over a noisy channel using suitable encoding and decoding procedures to reduce the likelihood of error. However, he never proved how this is attainable practically. The encoding here is done by adding redundancy to the information symbols, so it's all about sensibly adding redundancy so that the actual message can be recovered properly.

In our day-to-day life, we observe many noises in communication channels. The noise here means we do not receive what was communicated, also known as an error. For example, in deep space communication, the satellite is the message source, the outer space with the hardware that sends and receives messages is the channel, and the station on the earth is the receiver. The noise, in this case, may arise due to thermal disturbance. In a compact disc, picture, video, audio, or any data stored in the disc is a message. Disc itself is the channel and the viewer/listener is the receiver, here a noise may arise due to some scratches or fingerprints on the disc. Some other examples of communication channels are telephone, atmosphere, storage devices like HDD, SSD, floppy, RAM, pen drive, etc. The main objective of coding theory is to decode a received message correctly based on what is transmitted.

Shannon worked in an era when computers were widely used for large calculations. During this, the users were facing some errors. The computers then were capable of detecting the errors but not correcting them. This led to the necessity of configuring an algorithm capable of doing both things simultaneously. This problem was solved to a large extent by the Hamming codes [40], named after an American mathematician, Richard Wesley Hamming. This work was also carried out at the Bell Laboratories. The discovery of this class of codes is a result of the rigorous work done by him on automatic error-corrections on punched card readers. Hamming was the first person in history to develop error-correcting codes.

On computer networks, there are two types of error-correcting codes: block codes and convolutional codes. Block codes are a family of error-correcting codes where data is encoded in blocks. A block code is called a linear code if it is a subspace of the vector space $\mathbb{F}_q^n$ over a finite field $\mathbb{F}$, otherwise it is non-linear. A linear code is represented as an (n, k) code, where n is the length of the code and k is the dimension of the code. If we include the minimum distance d of the code, it is written as (n, k, d) code. Block codes are memoryless, which means that one block is independent of the other blocks in an encoded message. Therefore, accurate frame synchronisation is required for decoding; frame synchronisation means that the decoder is aware of the position of symbols in a received codeword. On the other hand, convolutional codes [33] were introduced by Peter Elias in 1955. Here, the information symbols are not independent of the other symbols and are spread throughout the sequence. Unlike the block codes, these codes have memory. Our study is focused on block codes, so we discuss some major developments in the field of block codes.

Hamming codes are capable of correcting single errors and detecting double errors. In later stages, these codes turned out to be of great importance from a theoretical as well as practical perspective in the development of coding theory. Hamming codes are perfect, which means they have the highest possible code rate pertaining to the given parameters. Even though a general approach was elaborated by Hamming for this construction, he focused on the $(7, 4)$ Hamming code, where 4 information bits are transmitted by adding 3 parity check bits. Hamming introduced the concept of minimum distance between two codewords in a code; this concept follows the lines of a metric space, so has similar interesting properties. Shannon and Hamming's works were complementary to one another. Shannon's work was on a probabilistic approach, while Hamming focused on the combinatorial approach.

The development of Hamming codes was a significant step forward in coding theory. The codes were capable of correcting only one-bit error at a time. Then a contemporary Swiss-born mathematician, Marcel Jules Edouard Golay, was able to develop two codes capable of correcting more bits at a time [38]. These constructions were a generalisation of Hamming's constructions. First, he constructed a binary $(23, 12, 7)$ code, which was capable of correcting 3 bit errors at a time. Second, he constructed a ternary $(11, 6, 5)$ code capable of correcting 2 bit errors at a time. A fascinating fact about this paper is that, despite being barely one page in length, it brought a revolution in coding theory and it has deep connections to number theory, combinatorics, graph theory, game theory, group theory, and other similar fields.

The construction of error-correcting codes becomes easy due to their algebraic structure, Irving Stoy Reed [91], David Eugene Muller [65], and David Slepian's [103] development should be credited for this. Reed and Muller demonstrated the use of finite fields and rings in the construction of error-correcting codes, whereas Slepian demonstrated group codes. After the code construction was done by Muller [65], Reed [91] shortly developed the decoding algorithm where up to half the minimum distance decoding was possible. Together, this class of codes is known as the Reed-Muller (RM) codes. This construction was a step forward in the development of error-correcting codes as the flexibility for code size and the number of errors possible to be corrected in a codeword was higher than the Hamming and Golay codes. The Mariner mission to Mars in 1965 was able to successfully capture black and white images of the surface there. The quality of pictures improved with the Mariner 9 in 1972. The reason behind this is the use of RM codes for the transmission, where 6 bits of information were encoded by adding 26 more bits [116]. Between 1969 and 1977, RM codes were a great choice for communication scientists. However, this trend diminished with the construction of better codes. RM codes are being examined again in optical communications due to their high-speed decoding algorithm.

Another major contribution to error-correcting codes is the cyclic codes [75] introduced by Eugene August Prange in 1957 at the Air Force Cambridge Research Laboratory in Massachusetts. This is a class of linear block codes where the cyclic shift of any codeword is again a codeword. This algebraic property turned out to be a great choice for designing the codes smoothly. A generator polynomial of degree $n-k$ is used to generate an (n,k) cyclic code; these codes are also known as cyclic residue check (CRC) codes. The Meggit decoder [62] can be used to decode this class of codes. Since the complexity of the Meggit decoder increases exponentially as the number of correctable erroneous bits increases, the use of CRC codes is confined to single and double bit correction. These days, CRC codes are mainly used in error detection rather than correction. Parallel to the construction of the cyclic codes, Alexis Hocquenghem [43] in 1959 and Raj Chandra Bose and Dwijendra Kumar Ray-Chaudhuri [21] in 1960 found a subclass of cyclic codes. Together, these codes are known as the BCH codes of length $n = q^m - 1$, where q is the order of the field and m is a positive integer. A binary (n,k) BCH code is capable of correcting at least $\frac{n-k}{m}$ errors. The extension of the BCH codes for non-binary cases was done by Irving Stoy Reed and Gustave Solomon in 1960, known as the RS codes [92]. The non-binary nature of the code made it suitable for the correction of burst errors. However, these codes were considered for extensive applications after Elwyn Ralph Berlekamp [12] developed an efficient decoding algorithm for RS codes in 1967. Since then, these codes have been frequently used in VCD players, DVD players, and other devices [14].

Other than the construction of codes over finite fields, Hamming was also interested in knowing the maximum possible number of codewords in a code of length n with minimum distance d. This problem gave rise to the Sphere Packing Bound, also known as the Hamming Bound [40]. Here, an upper bound on the number of codewords is derived for codes having minimum distance d and length n defined over a Galois field. Any code attaining Hamming bound is called a **perfect code**. Following it, in coding theory, many upper and lower bounds on the number of codewords have been found. The first lower bound on the number of codewords with a fixed length and minimum distance is due to Edgar Nelson Gilbert [37] and Rom Rubenovich Varshamov [113], commonly known as Gilbert-Varshamov bound.

Gerald Enoch Sacks [95] further gave a much simpler proof of this bound. Plotkin bound by Moriss Plotkin [72] further improved the Hamming bound, which is valid if the minimum distance d is close to the code length n.

James H. Griesmer developed the Griemer bound [39] on the length of the code for linear codes, which determines the existence of a linear code. Later on, Solomon and Stiffler [104] and Belov [10] found simplex codes satisfying the Griesmer bound. The Singleton bound is a very simple upper bound developed by Robert C. Singleton [102]. This bound leads to the class of Maximum Distance Separable (MDS) codes, which contains the RS-codes. Assmus, Mattson and Turyn [7], Forney [35] and Kassami, Lin and Peterson [50] further carried out this study independently. S. Reiger developed bounds for codes detecting and correcting bursts simultaneously, called the Reiger bound [93]. Some of these results were similar to the Fire bound [34]. Later, Campopiano [24] got a bound for linear codes correcting single burst, called the Campopiano bound. Following Plotkin's approach, S. Johnson [47] developed restricted and unrestricted upper bounds on the number of codewords for constant weight codes. Using these bounds, the Johnson upper bound was found.

Some review articles which are worth mentioning in this direction are Kautz and Levitt [51], Wolf [116], Assmus and Mattson [6], van Lint [109], Valenti [107], Dass and Das [29], etc. Some noteworthy books in coding theory in chronological order can be mentioned as - Peterson [69], Abramson [2], Gallagher [36], Berlekamp [13], van Lint [108], Peterson and Weldon [70], Blake [16], Blake and Mulin [18], McEliece [61], Clark (Jr) and Cain [26], MacWilliams and Sloane [60], Lin and Costello [58], Hill [42], Rhee [94], Vanschot and Oorschot [111], Poli, Hugnet and Craig [73], Vermani [114], Pless [71], Baylis [9], Lee [56], Morelas-Zaragoza [64], Huffman and Pless [45], Justesen and Hoholdt [49], Ling and Xing [59], van Lint [110], Klove [53], Neubauer, Freudenberger and Kuhn [66], Bose [20], etc.

The codes discussed so far were defined mainly over finite fields $\mathbb{F}_q$; codes are also studied over the ring $\mathbb{Z}_p$ (p, a prime number) [17]. By doing so, the codes worked fine, but several interesting problems from a ring theoretic point of view remained unsolved in popular codes like the BCH codes [17]. It was found that the codes defined over integer residue rings were more useful for use in computer-to-computer communications compared to the traditional codes defined over finite fields (having non-prime order) [17]. Ease in the arithmetic among integers can be considered a reason for this. Mathematicians became interested in researching codes over rings as a result of this. Early work in this regard is due to Blake ([17], [15]) and Spiegel [105]. Based on the work of Varshamov and Tenengolz [112] and Levenshtein and Vinck [57], in 1998, Vinck and Morita [115] studied a class of codes defined over the ring of integers, $\mathbb{Z}_m$, termed as **integer codes**.

1.2 Integer codes

Integer codes are a class of codes defined over the ring of integers modulo m. The inception of defining codes exclusively over the ring of integers modulo m began with the codes defined by Ian F. Blake [15] in 1972. He constructed cyclic codes over the ring $\mathbb{Z}_m$, where m is a product of distinct primes. Continuing this, in 1975, Blake [17] again constructed another class of codes over $\mathbb{Z}_m$, where m is a power of prime. In 1977, Eugene Spiegel [105] generalised this concept with the help of ring isomorphisms and constructed codes for any random value of m.

Based on the early work of Varshamov and Tenengolz [112] and from the results of perfect (d, k) codes [57] that corrected single peak-shifts, Vinck and Morita [115] gave us a class of codes over the ring $\mathbb{Z}_m$, which is today popularly known as the integer codes. These codes are suitable for channels where symbols can be represented by integers. This is mainly observed in magnetic recording and coded modulation. In discussing the applications of integer codes, single peak-shifts, single square errors, single cross errors, etc. correcting codes are demonstrated in [115] for magnetic recording systems. They used computer search results to construct the parity check matrix for the integer codes capable of correcting these errors.

Coded modulation is a strategy that effectively combines coding and modulation techniques. Each point in the signal constellation represents a symbol from the ring $\mathbb{Z}_m$ in block coded modulation. Thus, the information symbols are mapped to the integers in $\mathbb{Z}_m$ and the error-correcting procedure is carried out over $\mathbb{Z}_m$. In integer codes, possible error patterns occurring in several channels are identified and a parity check matrix is constructed for developing the code capable of correcting these prevalent errors. Single ± 1 error-correcting integer codes can be found in [55], where the ring considered is of the type $\mathbb{Z}_{2^l}$. A general construction for codes over the rings of the types $\mathbb{Z}_m$ with m equal to $p, 2p, pq, p^k$ and t^{k+1} are discussed in [54] (here p and q are primes and $t \in \mathbb{N}$). The codes are capable of correcting single $(\pm e_1, \pm e_2, \dots, \pm e_s)$ errors; in a particular case, $(\pm e_1, \pm e_2, \dots, \pm e_s) = (\pm 1, \pm t, \dots, \pm t^{k-1})$ is considered.

Another class of integer codes is constructed over the rings of the type $\mathbb{Z}_{2^b-1}$ by Radonjic and Vujicic [81]. The type of error considered here is a burst, which may occur within a b-bit byte. Unlike the traditional way, the symbols are first converted into binary b-tuples and then the error-correcting procedure is carried out. The error-correcting procedure could be efficiently carried out since a non-zero integer in the ring $\mathbb{Z}_{2^b-1}$ can be uniquely represented in its binary form. By converting the symbols into binary form tuples, these classes of integer codes become suitable for implementation in channels where asymmetric errors are evident.

Continuing the study on integer codes over the ring $\mathbb{Z}_{2^b-1}$, Radonjic and Vujic have contributed immensely in this area (see next paragraph). They have also shown the implementation of these codes in multi-core processors (quad-core, octa-core, dual-core, etc.). The encoding procedure in this class is based on a predetermined parity check matrix consisting of coefficients C_i, which are found by suitable computer search results. For decoding, look-up table operations are used. Due to this, very little memory is consumed during the process.

Integer codes correcting double asymmetric errors [89], spotty byte asymmetric errors [82], random and asymmetric errors [85], high-density byte asymmetric errors [83], single errors and burst asymmetric errors [84], sparse byte errors [86], burst asymmetric and double asymmetric errors [87], single errors (perfect) [77], double errors and triple adjacent errors [78], single errors and detecting burst errors [79], etc. are some of the major contributions in this direction.

1.3 Some specific error patterns

Developing various approaches in coding theory is nothing more than a contribution to the code's ability to detect and correct errors. Every time a new error-correcting technique is developed, it is attempted to increase the efficiency of the code while also pursuing better encoding and decoding methods. During transmission, errors may occur anywhere, randomly and independently of each other. These types of errors are called "random errors". In deep space, satellite channels, etc., these errors are prevalent. For Additive White Gaussian Channel (AWGN) channels, many random error-correcting schemes have been developed [101].

The occurrence of errors is entirely dependent on the communication channel's nature. Messages are generally sent over a channel in the form of a long string of signals and are placed one after another. During transmission, due to noise, the signals may be erased, faded, or altered. This will lead to an erroneous form of the message at the receiver's end. Sometimes, the message may be erased, faded, or altered in consecutive positions. Abramson and Elspas [1] pioneered the development of consecutive error-correcting codes after Hamming's code [40] on single error. He presented a class of codes capable of correcting single and double consecutive errors. Generalising this, Fire [34] discovered a general type of error which occurred in consecutive positions in the form of vectors and termed these types of errors "burst errors". In the report submitted to the Sylvania Reconnaissance Systems Laboratory, Fire [34] defined "open-loop bursts" as follows.

Definition 1.1. [34] *An open loop burst (or simply burst) of length l is an n-tuple in which non-zero components are confined to some l consecutive positions, the first and last components of which are necessarily non-zeros.*

$(1, 1, 0, 0, 0, 0, 0, 0), (0, 0, 1, 0, 1, 0, 1, 0), (1, 1, 1, 1, 0, 0, 1, 0)$ are examples of (open loop) bursts of length $2, 5$ and 7 respectively.

Performance of burst error-correction codes can be improved by using interleaving techniques in the scheme [44]; interleaving means rearrangements in the code. After the discovery of burst errors, researchers continued this to find other types of bursts occurring in different communication channels under various circumstances. In 1965, Chien and Tang [25] identified some channels where it is not necessary for a burst error to have the last component non-zero. This type of burst is called CT-burst.

Definition 1.2. [25] *A CT-burst of length l is an n-tuple having non-zero components confined to l consecutive positions, first of which is non-zero.*

For example, $(0,0,0,1,0,1,0,0)$ can be considered as a CT-burst of length $3,4$ as well as 5. It has been discovered that CT-bursts are useful for analysing errors in the experiments of telephone lines [3]. Dass [28] in 1980 modified the definition of CT-bursts for channels and named it as burst of length l (fixed). In the modified definition, Dass considered the bursts that do not occur after the $(n-l+1)^{th}$ position.

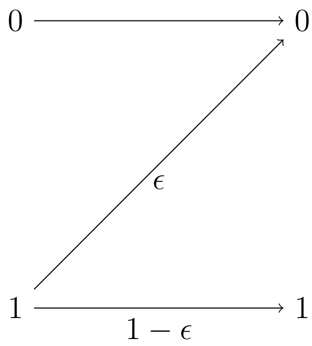
For a low-intensity burst, the number of erroneous components inside the bursts is very low. This was observed by Wyner [117] in 1963, and he named such bursts as low-density bursts. For high-intensity burst, the number of erroneous components inside the burst is very high. Such bursts are called high-density bursts ([22], [11]). So, applying the usual error-correcting techniques will lead to complexity and excess memory consumption in both of the types discussed above.

In the same way, we can reduce the undesirable complexity by separating CT-bursts into low-density and high-density types. The code's efficiency improves as a result of this. Solid burst is another type of burst that is prevalent in supercomputer storage systems [5], semiconductor memory data [46], etc. Here, all components inside the burst length are non-zeros, this type of burst is also known as an adjacent burst. The codes developed for correcting this class of bursts are mainly double and triple adjacent error-correcting.

Definition 1.3. [100] *A solid burst of length l is a burst of length l in which all the l components are non-zeros.*

For example, the bursts $(1,1,1,1,1,0,0,0)$ and $(0,1,1,1,1,1,1,0)$ are solid bursts of length 5 and 6 respectively, whereas $(0,0,1,0,1,1,1,0)$ is not a solid burst as there

Figure 1.1: **Z-channel with crossover probability ϵ**



exists a component 0 within length 5. However, the latter error can be considered as a burst of length 5 and a CT-burst of length 5 or 6.

In binary-oriented codes, possible error patterns can arise due to $1 \to 0$ and $0 \to 1$. It depends on the channel what the probability of $1 \to 0$ and $0 \to 1$ is. In some practical systems, the occurrence of $1 \to 0$ is extremely high [85]. Optical networks without optical amplifiers is an example regarding this [90]. We consider Z-channel (Figure 1.1) for binary asymmetric channels (BAC) where the probability of $0 \to 1$ is zero. Therefore, we redefine bursts, CT-bursts, low-density and high-density CT-bursts, solid bursts for Z-channel as the respective bursts that follow only the pattern $1 \to 0$.

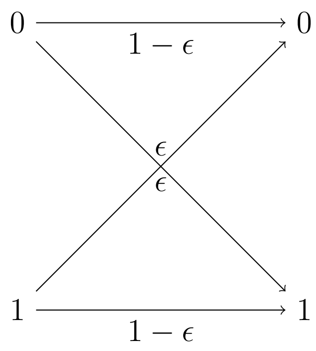
Definition 1.4. *A binary oriented burst where only* $1 \to 0$ *is a possibility of error is called an asymmetric burst.*

Definition 1.5. *A binary oriented CT-burst where only* $1 \to 0$ *is a possibility of error is called an asymmetric CT-burst.*

Definition 1.6. *A binary oriented solid burst where only* $1 \to 0$ *is a possibility of error is called an asymmetric solid burst.*

For example, $(0, 0, 0, 1, 1, 0, 1, 0)$ can be considered as an asymmetric CT-burst of length 4 or 5, also it can be considered as an asymmetric burst of length 4. $(1, 1, 1, 1, 0, 0, 0, 0)$ is an example of asymmetric solid burst of length 4. In the case of a low-density CT-burst, we specify the weight w (in Hamming sense) and consider

Figure 1.2: **Binary symmetric channel**



the CT-bursts that have a weight up to w. Similarly, for high-density, we consider CT-bursts that have a weight of at least w.

Definition 1.7. *A low-density asymmetric CT-burst ($LACTB_{d/l}$) is an asymmetric CT-burst of length l in which the number of inverted (erroneous) bits d is between* 1 *and* $\lfloor \frac{l}{2} \rfloor$, $1 \leq d \leq \lfloor \frac{l}{2} \rfloor$.

Definition 1.8. *A high-density asymmetric CT-burst ($HACTB_{h/l}$) is an asymmetric CT-burst of length l in which the number of inverted (erroneous) bits h is between* $\lceil \frac{l}{2} \rceil$ *and* l, $\lceil \frac{l}{2} \rceil \leq h \leq l$.

$(1, 0, 0, 1, 0, 0, 0, 0)$ and $(1, 0, 1, 0, 1, 0, 0, 0)$ are examples of low-density and high-density CT-bursts of length 4 and 5, with weight 2 and 3 respectively.

In some VLSI circuits, both of the patterns $1 \to 0$ and $0 \to 1$ may occur, but not both at the same time. Unidirectional errors occur as a result of this ([74], [4]). This can be modelled with the help of a binary symmetric channel (BSC)(Figure 1.2), where occurrences of both $1 \to 0$ and $0 \to 1$ are equally likely with the crossover probability ϵ. Thus, we define unidirectional solid bursts as following:

Definition 1.9. *A binary-oriented solid burst where errors occur either in the pattern* $1 \to 0$ *or* $0 \to 1$, *but not simultaneously, is called a unidirectional solid burst.*

The likelihood of the pattern $1 \to 0$ or $0 \to 1$ occurring is not known in advance. For example, $(1, 1, 1, 0, 0, 0, 0, 0)$ and $(0, 0, 0, -1, -1, -1, -1, 0)$ are unidirectional solid bursts of length 3 and 4 respectively. But, the solid burst $(0, 0, 0, -1, 1, 1, 0, 0)$ will not be unidirectional.

So far, we have discussed asymmetric or unidirectional burst/CT-burst (low and high-density)/solid burst errors that are studied in this thesis. Other than these types of errors, the following error types are also used in our study.

Definition 1.10. [86] *Sparse byte error is a type of error where the bits are distorted in one, two, or three adjacent positions.*

Definition 1.11. [82] *A t-spotty byte error is an error occurring at t random positions. Further, the errors following an asymmetric pattern, $1 \to 0$, are called t-spotty byte asymmetric errors.*

For example, $(1, 0, 0, 0, 0, 0, 0, 0)$, $(0, 0, 0, 0, -1, -1, 0, 0)$, $(-1, 1, -1, 0, 0, 0, 0, 0)$ are sparse byte errors and $(1, 1, 0, 0, 0, 1, 0, 0)$ is a 3-spotty byte asymmetric error.

1.4 Previous results on integer and linear codes

In this section, we briefly discuss the preliminary results and ideas that are used in our study. The definition of integer codes is given by Vinck and Morita [115] as follows.

Definition 1.12. [115] *Let $m, M, N \in \mathbb{N}$, $H \in \mathbb{Z}_m^{M \times N}$ and $d \in \mathbb{Z}_m^M$. The integer code is defined by $\{a \in \mathbb{Z}_m^N : aH^T = d, H^T$ is the transpose of matrix $H\}$. Here H is the **check matrix** for the integer code.*

Note: Without the loss of generality, we may assume that $d = 0$. An integer code with $d \neq 0$ can also be transformed into an integer code with $d = 0$ by subtracting one codeword from all other codewords of the code. The following result is important for the enumeration of the number of different codes.

Result 1.13. [115] *If the greatest common divisor (gcd) of the $M \times M$ subdeterminants of a matrix H is equal to a unit in $\mathbb{Z}_m$, then the check matrix H defines m^M different integer codes of equal size m^{N-M}.*

Definition 1.14. [115] *Let C be an integer code with check matrix H and $d = 0$. Then the **syndrome** S of a received word $r \in \mathbb{Z}_m^N$ is defined by $S = rH^T = eH^T$, where e is the error vector.*

Definition 1.15. [115] *An integer code is called s-error-correcting integer code of error size t if all errors $e = (e_1, e_2, \ldots, e_N)$ with weight$(e) \leq s$ and $e_i \in \{-t, -t+1, \ldots, t-1, t\}$ for all i can be corrected, here weight(e) is the number of non-zero components in e.*

In relation to the codes discussed above, Strocks [106] proved the following two results.

Result 1.16. [106] *For the check matrix $H = (h_1, h_2, \ldots, h_N)$ of a perfect integer code with $M = 1$ and $s = 2$, the following must hold:*

- $\forall i\ \exists\ j \neq i$ *and* $\exists\ A \in \{-t, \ldots, t\}$*:* $(Ah_j = (2t+1)h_i)$.
- $\forall i$*,* $\forall\ j \neq i$ *and* $\forall\ A \in \{-t, \ldots, t\}$*:* $(h_j \neq Ah_i)$.

Using the results above and similar approaches, some error-correcting codes are discussed in [55] and [54]. Other similar works can be found in their references. Using the concept of integer codes based on a predetermined parity check matrix, Radonjic and Vujicic [81] introduced another form of integer codes over the ring $\mathbb{Z}_{2^b-1}$. In this class, k information integers are converted into k b-bit bytes by using the binary representation of the information integers. They derived a number of results for channels having asymmetric errors in different ways. Some of the error patterns, results, and encoding/decoding techniques in this regard that will be helpful in our study have been mentioned below.

Let $B = (x_{b-1}, x_{b-2}, \ldots, x_0)$ be the sent and $\bar{B} = (x'_{b-1}, x'_{b-2}, \ldots, x'_0)$ be the received vector affected due to the t-bit burst $(0 < t < b)$ (refer Definition 1.1). Then, error $e = B - \bar{B} = 2^r(c_{r+t-1}2^{t-1} + c_{r+t-2}2^{t-2} + \ldots + c_0 2^0)$,
$0 \leq r \leq b-t$ and $c_{r+j} = x_{r+j} - x'_{r+j} = \begin{cases} \pm 1, & j = 0, t-1 \\ 0, \pm 1, & j = 1, 2, \ldots, t-2. \end{cases}$
It is obvious that $e = \pm 2^r m$, m is odd and $1 \leq m \leq 2^t - 1$.

Definition 1.17. [81] *The set of all t-bit bursts occurring within a b-bit byte is* $e_{b,t} = \{\pm 2^r(1, 3, 5, \ldots, 2^t - 1) : \ r = 0, 1, 2, \ldots, b - t\}$.

Definition 1.18. [81] *The set of all bursts up to length l occurring within a b-bit byte is defined by* $\epsilon_{b,l} = e_{b,1} \cup e_{b,2} \cup \ldots \cup e_{b,l}$, *i.e., union of all bursts upto length l.*

Result 1.19. [81] *The cardinality of the set $\epsilon_{b,l}$ is* $|\epsilon_{b,l}| = 2^l(b - l + 2) - 2$.

Using the definitions and result above, integer $(B_lEC)_b$ codes are developed, capable of correcting burst up to length l. This is done by finding coefficients from the ring $\mathbb{Z}_{2^b-1}$ satisfying the condition given below.

Definition 1.20. [81] *The syndrome set for $((k+1)b, kb)$ integer $(B_lEC)_b$-codes is defined by* $\zeta_{b,l} = \bigcup\limits_{i=1}^{k+1} \left\{C_i\epsilon_{b,l} \pmod{2^b - 1}\right\}$, *where the coefficients* $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0, 1\}$, *for $i = 1, 2, \ldots, k$ are chosen in such a way that each C_i multiplied with each element of $\epsilon_{b,l} \pmod{2^b - 1}$ yields a different result and $C_{k+1} = -1$.*

For encoding purpose, a new b-bit byte called **check byte** C_B is calculated as $C_B = [C_1B_1 + C_2B_2 + \ldots + C_kB_k] \pmod{2^b - 1}$. The decoder constructs a look up table by using the elements of the syndrome set and further categorize the elements as $\epsilon_{b,l}$ and $C_i\epsilon_{b,l}$, $\epsilon_{b,l}$ pertains to an error in the check byte, whereas $C_i\epsilon_{b,l}$ pertains to an error in the i^{th} data byte.

Suppose a transmitted message $B_1B_2 \ldots B_kC_B$ is received as $\bar{B_1}\bar{B_2} \ldots \bar{B_k}\bar{C_B}$, then the decoder calculates syndrome $S = C_{\bar{B}} - \bar{C_B} \pmod{(2^b - 1)}$. If $S = 0$, then the message is considered to be free of error; if $S \neq 0$ and is available in the look up table, the decoder will be able to decode it, else it is beyond the scope of the decoder.

In asymmetric burst, the error is of $1 \to 0$ type. Thus, the pattern of possible asymmetric burst errors will be $e_{r,l} = -2^r(2^{l-1} + a_{l-2}2^{l-2} + \ldots + a_12 + 2^0)$, where $0 \leq r \leq b - l$ and $a_j \in \{0, 1\}$ for $1 \leq j \leq l - 2$. In [88], error-correcting method for integer codes correcting asymmetric burst errors within a byte is done similar to the symmetric burst errors [81].

Definition 1.21. [88] *Let* $0 \leq r \leq b - l$, $3 \leq m \leq 2^l - 1$, *where m is odd and let* $e_{r,m} = 2^rm$ *be the difference between the integer values of the correct b-bit byte and* *its erroneous counterpart affected by the asymmetric burst error of length l within the b-bit byte. Then, the set of syndromes corresponding to the asymmetric burst errors is defined as*

$$S = \left\{ \bigcup_{r=0}^{b-l} \bigcup_{m=3}^{2^l-1} \bigcup_{i=1}^{k+1} (-e_{r,m}C_i) \pmod{(2^b-1)} \right\}.$$

In the definition above, the coefficients $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0,1\}$ for $i = 1, 2, \ldots, k$ are chosen in such a way that each C_i multiplied by each element of $-e_{r,m} \pmod{2^b-1}$ yields a different result.

Result 1.22. [88] *The code defined above can correct all asymmetric burst errors of length $2 \leq l \leq b-1$ if and only if there exist k coefficients $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0,1\}$ such that $|S| = [2^{l-1}(b-l+2)-1](k+1)$.*

In [89], integer codes correcting double asymmetric errors (IDAEC codes) are presented. Here, a single asymmetric error of the type $1 \to 0$ is considered simultaneously in two data bytes, or one in a data byte and another in the check byte, or a single error in a data byte or check byte. After considering the error pattern in single asymmetric error, the set of syndromes is as follows:

$s_1 = \Big\{2^r \pmod{(2^b-1)} : 0 \leq r \leq b-1\Big\}$, error in the check byte,

$s_2 = \Big\{ \bigcup_{i=1}^{k} (-2^r C_i) \pmod{(2^b-1)} : 0 \leq r \leq b-1\Big\}$, error in i^{th} data byte,

$d_1 = \Big\{(2^r + 2^s) \pmod{(2^b-1)} : 0 \leq r < s \leq b-1\Big\}$, both errors in the check byte,

$d_2 = \Big\{ \bigcup_{i=1}^{k} [(-2^r - 2^s)C_i] \pmod{(2^b-1)} : 0 < r < s \leq b-1\Big\}$, both errors in the same i^{th} data byte,

$d_3 = \Big\{ \bigcup_{i=1}^{k} [-2^r C_i + 2^s] \pmod{(2^b-1)} : 0 \leq r, s \leq b-1\Big\}$, first error in i^{th} data byte and another in the check byte,

$d_4 = \Big\{ \bigcup_{i=1}^{k-1} \bigcup_{j=i+1}^{k} [(-2^r C_i - 2^s C_j)] \pmod{(2^b-1)} : 0 \leq r, s \leq b-1\Big\}$, first and second error in i^{th} and j^{th} data byte respectively.

Result 1.23. [89] *The IDAEC code can correct all single and double asymmetric errors if and only if there exist k mutually different coefficients $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0,1\}$ such that*

$|s_1| = b$

$|s_2| = bk$

$|d_1| = \frac{b}{2}(b-1)$

$|d_2| = \frac{b}{2}(b-1)k$

$|d_3| = b^2k$

$|d_4| = \frac{b^2}{2}(k-1)k$

$s_1 \cap s_2 \cap d_1 \cap d_2 \cap d_3 \cap d_4 = \phi.$

Result 1.24. [89] *The cardinality of the syndrome set of a* $((k+1)b, kb)$ *integer IDAEC code is* $\frac{b}{2}(k+1) + \frac{b^2}{2}(k+1)^2$.

The results mentioned below are for integer codes correcting spotty byte asymmetric errors.

Definition 1.25. [82] *The set of syndromes corresponding to* t*-spotty byte asymmetric error within a* b*-bit byte is* $S = \left\{ \bigcup_{m=1}^{t} \bigcup_{i=1}^{k+1} (-C_i e_m) \pmod{2^b - 1} : 1 \leq t < b \right\}$, *where* $e_m = \{2^{x_1} + 2^{x_2} + \ldots + 2^{x_m}\}$ *with* $0 \leq x_1 < \ldots < b$ *and* $1 \leq m \leq t$.

Result 1.26. [82] *An integer code can correct all* t*-spotty byte asymmetric errors if and only if there exist* k *mutually distinct coefficients* $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0, 1\}$ *such that* $|S| = (k+1) \sum_{m=1}^{t} \binom{b}{m}$.

The results mentioned below are for sparse byte errors.

Definition 1.27. [86] *The syndrome set for an integer code correcting sparse byte within a* b*-bit byte will be* $S = \epsilon_1 \cup \epsilon_2 \cup \epsilon_3$, *where*

$\epsilon_1 = \left\{ \bigcup_{i=1}^{k+1} (\pm 2^r C_i) \pmod{2^b - 1} : 0 \leq r \leq b-1 \right\}$,

$\epsilon_2 = \left\{ \bigcup_{i=1}^{k+1} [(\pm 2^r \pm 2^s) C_i] \pmod{2^b - 1} : 0 \leq r < s \leq b-1 \right\}$,

$\epsilon_3 = \left\{ \bigcup_{i=1}^{k+1} [(\pm 2^0 \pm 2^1 \pm 2^2) 2^m C_i] \pmod{2^b - 1} : 0 \leq m \leq b-3 \right\}$.

Result 1.28. [86] *An integer code can correct all sparse byte errors if and only if there exist* k *mutually distinct coefficients* $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0, 1\}$ *such that* $|S| = (k+1)\left[2(b-1)^2 - 2\right]$.

In [78], integer codes correcting double errors within and across two b-bit bytes and triple adjacent errors within a b-bit byte are discussed. The results pertaining to the codes are given below:

Definition 1.29. [78] *The set of syndromes belonging to double errors is given by* $S_1 = \epsilon_1 \cup \epsilon_2$, *where*

$$\epsilon_1 = \left\{ \bigcup_{i=1}^{k+1} (\pm 2^r \pm 2^s)\, C_i \pmod{2^b - 1} \,:\, 0 \leq r < s \leq b-1 \right\},$$

$$\epsilon_2 = \left\{ \bigcup_{i=1}^{k} \bigcup_{j=i+1}^{k+1} (\pm 2^r C_i \pm 2^s C_j) \pmod{2^b - 1} \,:\, 0 \leq r, s \leq b-1 \right\}.$$

Definition 1.30. [78] *The set of syndrome for triple adjacent errors occurring within a b-bit byte will be*

$$\epsilon_3 = \left\{ \bigcup_{i=1}^{k+1} (\pm 2^2 \pm 2^1 \pm 2^0)\, 2^r C_i \pmod{2^b - 1} \,:\, 0 \leq r \leq b-3 \right\}.$$

By combining the syndrome sets above, we get the syndrome set (S) pertaining to both the errors.

Result 1.31. [78] *An integer code can correct double errors (throughout) and triple adjacent errors within a b-bit byte if and only if there exist k mutually distinct coefficients* $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0, 1\}$ *such that* $|S| = 2\left[b(k+1) - 1\right]^2 - 2$.

Whenever double errors are in two different bytes, technique from [89] is followed, and for triple adjacent errors in a b-bit byte, technique from [81] is followed. In [87], integer codes correcting asymmetric burst errors within a b-bit byte and double asymmetric errors throughout the codeword have been presented. The results for the same are as follows:

Definition 1.32. [87] *The set of syndromes corresponding to asymmetric burst errors occurring within a b-bit byte is* $\epsilon_1 = S_1 \cup S_2$, *where*

$$S_1 = \left\{ -2^r(2m-1)C_i \pmod{2^b - 1} :\, 0 \leq r \leq b-l, 1 \leq m \leq 2^{v-1}, 1 \leq v \leq l, 1 \leq i \leq k \right\},$$

$$S_2 = \left\{ 2^r(2m-1) \pmod{2^b - 1} : 0 \leq r \leq b-l, 1 \leq m \leq 2^{v-1}, 1 \leq v \leq l \right\}.$$

Definition 1.33. [87] *The set of syndromes corresponding to double asymmetric errors occurring within a b-bit byte is* $\epsilon_2 = S_3 \cup S_4$, *where*

$$S_3 = \left\{ -2^r(2^{s-r} + 1)C_i \pmod{2^b - 1} : 0 \leq r < s \leq b-1, 1 \leq i \leq k \right\},$$

$$S_4 = \left\{ 2^r(2^{s-r} + 1) \pmod{2^b - 1} : 0 \leq r < s \leq b-1 \right\}.$$

Definition 1.34. [87] *The set of syndromes corresponding to double asymmetric errors occurring in two b-bit bytes is* $\epsilon_3 = S_5 \cup S_6$, *where*

$$S_5 = \left\{ -2^r C_i - 2^s C_j \pmod{2^b - 1} :\, 0 \leq r, s \leq b-1, 1 \leq i < j \leq k \right\},$$

$$S_6 = \left\{ -2^r C_i + 2^s \pmod{2^b - 1} :\, 0 \leq r, s \leq b-1, 1 \leq i \leq k \right\}.$$

Definition 1.35. [87] *The set of syndromes corresponding to double asymmetric errors excluding the asymmetric burst errors is* $\epsilon_4 = S_7 \cup S_8$*, where*
$S_7 = \Big\{-2^r(2^{s-r}+1)C_i \pmod{2^b-1} : l \leq r+l \leq s \leq b-1, 1 \leq i \leq k\Big\}$,
$S_8 = \Big\{2^r(2^{s-r}+1) \pmod{2^b-1} : l \leq r+l \leq s \leq b-1\Big\}$.

Let S be the set of all the syndromes discussed above. Result below discusses the number of syndrome elements.

Result 1.36. [87] *A* $((k+1)b, kb)$ *integer code can correct asymmetric burst errors within a b-bit byte and double asymmetric errors throughout the codeword if and only if there exist k mutually distinct coefficients* $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0,1\}$ *such that* $|S| = (k+1)\left[\frac{2^l(b-l+2)+b^2k+(b-l+1)(b-l)-2}{2}\right]$.

In [85], integer codes are constructed that are capable of correcting asymmetric burst and random asymmetric errors of length l and t respectively, within a b-bit byte.

Definition 1.37. [85] *Let* $e_1 = \{1\}$ *and* $e_w = \{2^{w-1}+1, 2^{w-1}+1, \ldots, 2^w-1\}$*, then the set of syndromes corresponding to l/b asymmetric burst errors occurring within a b-bit byte is defined by* $\epsilon_1 = S_1 \cup S_2$*, where*
$S_1 = \left\{\bigcup_{m=1}^{l} (2^r e_m) \pmod{2^b-1} :\ 0 \leq r \leq b-m\right\}$,
$S_2 = \left\{\bigcup_{m=1}^{l} \bigcup_{i=1}^{k} (-C_i 2^r e_m) \pmod{2^b-1} :\ 0 \leq r \leq b-m\right\}$.

Definition 1.38. [85] *Let* $f_2 = \{2^s+2^z\}$ *and* $f_v = \{2^s+2^{x_1}+2^{x_2}+\ldots+2^{x_{v-2}}+2^z\}$*, then the set of syndromes corresponding to t/b random asymmetric errors within a b-bit byte is defined by* $\epsilon_2 = S_3 \cup S_4$*, where*
$S_3 = \left\{\bigcup_{n=2}^{l} (f_n) \pmod{2^b-1} :\ 2 \leq t < l\right\}$,
$S_4 = \left\{\bigcup_{n=2}^{t} \bigcup_{i=1}^{k} (-C_i f_n) \pmod{2^b-1} :\ 2 \leq t < l\right\}$.

By considering the union of the sets discussed above, the set of syndromes corresponding to asymmetric bursts and random asymmetric errors of length l and t respectively, will be $S = \epsilon_1 \cup \epsilon_2$.

Result 1.39. [85] *A* $((k+1)b, kb)$ *integer code can correct asymmetric bursts and random asymmetric errors of length l and t respectively, if and only if there exist k*

mutually distinct coefficients $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0,1\}$ *such that*
$|S| = (k+1)\left[2^{l-1}(b-l+2) - 1 + \frac{(b-l)^2+b-l}{2}\right]$ *for* $t = 2$,
$|S| = (k+1)\left[2^{l-1}(b-l+2) - 1 + \frac{(b-l)^2+b-l}{2}\right] - (k+1)\left[\frac{2(b-l)^3+3(b-l)^2+b-l}{6}\right]$ *for* $t = 3$.

In [84], integer codes are constructed that are capable of correcting single symmetric errors and asymmetric burst errors up to length l within a b-bit byte. Symmetric errors mean that the errors may occur as a result of both $1 \to 0$ and $0 \to 1$.

Definition 1.40. [84] *Let* $0 \le r \le b-1$ *and let* $e_r = \pm 2^r$ *be the difference between the integer values of the correct b-bit byte and its erroneous counterpart affected by single error. Then, the set of syndromes corresponding to single errors is defined as* $s_1 = \bigcup_{r=0}^{b-1} \bigcup_{i=1}^{k+1} e_r C_i \pmod{2^b - 1}$.

Definition 1.41. [84] *Let* $0 \le r \le b-l$, $3 \le m \le 2^l - 1$, *where* m *is odd, and let* $e_m = 2^r m$ *be the difference between the integer values of the correct b-bit byte and its erroneous counterpart affected l/b BA error. Then, the set of syndromes corresponding to l/b BA errors is defined as* $s_2 = \bigcup_{r=0}^{b-l} \bigcup_{m=3}^{2^l-1} \bigcup_{i=1}^{k+1} (-e_{r,m} C_i) \pmod{2^b - 1}$.

Result below determines the number of elements in the syndrome set of the integer code discussed in [84].

Result 1.42. [84] *The codes defined by Definition 1.40-1.41 can correct all single errors and all l/b BA errors if and only if there exist k mutually different coefficients* $C_i \in \mathbb{Z}_{2^b-1} \setminus \{0,1\}$ *such that*
$|s_1| = 2b(k+1)$
$|s_2| = (k+1)\left[2^{l-1}(b-l+2) - b - 1\right]$
$s_1 \cap s_2 = \phi$.

Apart from the integer codes, our study contains some linear and non-linear codes used for comparison. The existence and study of these codes are mentioned below.

Result 1.43. [68] *Any code that corrects an asymmetric/unidirectional burst of length b must have at least* $b + \log_2(k)$ *check bits, where k is the number of information bytes.*

Below are the necessary and sufficient conditions respectively, used for the existence of the linear codes discussed in [30].

Result 1.44. [30] *The number of parity check digits in an (n,k) linear code correcting all bursts of length l_1 (fixed) in the first block of length n_1, and all bursts of length l_2 (fixed) in the second block of length n_2 $(n = n_1 + n_2)$ is at least* $\log_q \Big[1 + \{(n_1 - l_1 + 1)q^{l_1-1} + (n_2 - l_2 + 1)q^{l_2-1}(q-1)\}\Big]$.

Result 1.45. [30] *Given positive integers l_1 and l_2, there exists an (n,k) linear code that corrects all bursts of length l_1 (fixed) in the first block of length n, and all bursts of length l_2 (fixed) in the second block of length $n_2 (n = n_1 + n_2)$ satisfying the inequality* $q^{n-k} > max\Big[q^{l_2-1}\{1+(n_2-2l_2+1)(q-1)q^{l_2-1}\},\ q^{l_1-1}\{1+(n_1-2l_1+1)(q-1)q^{l_1-1} + (n_2 - l_2 + 1)(q-1)q^{l_2-1}\}\Big]$.

Below are the necessary and sufficient conditions respectively, used for the existence of the linear codes discussed in [31].

Result 1.46. [31] *The number of parity check digits r required for an (n,k) linear code over $GF(q)$, sub-divided into s sub-blocks of length t each, that corrects burst of length l (fixed) with weight w or less lying within a single sub-block of length t is at least* $\log_q \Big[1 + s(t-l+1)(q-1)[1+q-1]^{(l-1,w-1)}\Big]$, *where* $[1+x]^{(n,m)}$ *is the incomplete binomial expansion of* $(1+x)^n$ *taken up to* m^{th} *terms.*

Result 1.47. [31] *An (n,k) linear code over $GF(q)$ capable of correcting bursts of length l (fixed) with weight w or less, $w \le l < \frac{t}{2}$, occurring within a single sub-block of length t can always be constructed using r check digits where r is the smallest integer satisfying the inequality* $q^r > [1+(q-1)]^{(l-1,w-1)}\Big[1+(q-1)[1+(q-1)]^{(l-1,w-1)}\{s(t-l+1)-l\}\Big] + \sum_{i=w}^{2w-1}\binom{l-1}{i}(q-1)^i + \sum_{k=1}^{l-1}\sum_{r_1,r_2,r_3}\binom{l-k}{r_1}\binom{k-1}{r_2}\binom{l-k-1}{r_3}(q-1)^{r_1+r_2+r_3+1}$, *where* $1 \le r_1 \le w-1, 0 \le r_2 \le 2w-3, 0 \le r_3 \le w-1, r_2 + r_3 \ge w-1, r_1 + r_2 + r_3 \le 2w-2$.

Below are the necessary and sufficient conditions respectively, used for the existence of the error locating codes discussed in [32].

Result 1.48. [32] *The number of check digits r required for an (n, k) linear code, subdivided into s sub-blocks of length t each, that locates a single corrupted sub-block containing errors that are bursts of length l (fixed) with weight w or more $(w \leq l)$ is bounded from below by* $r \geq \log_q \left[1 + s\left(q^{\lceil \frac{d}{2} \rceil} - 1\right)\right]$, $w \leq d \leq l$.

Result 1.49. [32] *A code capable of detecting burst errors of length l (fixed) with weight w or more $(w \leq l)$ occurring within a single sub-block, and of locating that sub-block, can always be constructed using r check digits where r is the smallest integer satisfying the inequality*

$$r > \log_q \left\{1 + \left[\sum_{i=w-1}^{l-1} \binom{l-1}{i}(q-1)^i\right]\left[1 + (s-1)(q-1)(t+l+1)\sum_{i=w-1}^{l-1} \binom{l-1}{i}(q-1)^i\right]\right\}.$$

Byte unidirectional error-correcting codes are discussed in [19], which are similar to [68]. Result below discusses the redundancy for determining code rate.

Result 1.50. [19] *Let m be the number of information bytes with b bits (1 byte). Then any code that corrects byte unidirectional errors needs at least* $\log(m(2^b - 1) + 1) \approx b + \log m$ *check bits.*

A class of linear codes correcting solid bursts is discussed in [27]. Let

$$A_k = \begin{bmatrix} x_1 & x_2 & . & . & . & x_{k-1} & x_k \\ 0 & x_2 & . & . & . & x_{k-1} & x_k \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & . & . & . & 0 & x_k \\ 0 & 0 & . & . & . & 0 & 0 \end{bmatrix}, Y = \begin{bmatrix} y \\ y \\ y \\ . \\ . \\ y \end{bmatrix}, \ B_k = \begin{bmatrix} 0 & 0 & . & . & . & 0 & 0 \\ x_k & 0 & . & . & . & 0 & 0 \\ x_k & x_{k-1} & . & . & . & 0 & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ x_k & x_{k-1} & . & . & . & x_2 & x_1 \end{bmatrix},$$

where $x_i, y \in \{1, 2, \ldots, q-1\}$, $(x_i, y) = 1$; $(x_i, x_j) = 1$ for $i \neq j$ and

$$H_k^t = \begin{bmatrix} A_k & Y & B_k \\ 0 & y & a_1 \\ 0 & y & a_3 \\ 0 & y & a_5 \\ . & . & . \\ . & . & . \\ 0 & y & a_t \end{bmatrix},$$

where a_t represents the k-tuple $(\underbrace{0,0,...,0}_{t},\ x_{k-t},\underbrace{0,0,...,0}_{t},\ x_{k-2t-1},\ldots,c)$ and

$$c = \begin{cases} x_1 & \text{if } \ k-t-1 \text{ is a multiple of } t+1 \\ 0 & \text{if } \ k-t-1 \text{ is not a multiple of } t+1. \end{cases}$$

Result 1.51. [27] *An* $(2k+1, k-\frac{t+1}{2})$ *linear code* C_k^t *over* $GF(q)$, t *is an odd number whose parity check matrix is* H_k^t, *is capable to correct*

1. *all solid bursts of length* $t+2$ *or less* $(t \leq 2k-3)$ *and all solid bursts of odd lengths upto* $2k-1$, *if* k *is odd,*
2. *all solid bursts of length* $t+2$ *or less* $(t \leq k-3)$ *and all solid bursts of odd lengths upto* $k-1$, *if* k *is even.*

1.5 Thesis layout

The thesis mainly studies integer codes that are capable of correcting certain types of error patterns. Different types of error patterns observed in communication channels are considered, and accordingly, the error-correcting algorithms are developed. The codes are constructed with the help of computer search results. Throughout our study, we have used Python software to determine the parity check matrix. A link to the source code for each of the error-correcting codes is attached in the appendices. The errors considered are either confined within a b-bit byte or spread across two b-bit bytes. The probability of erroneous decoding is derived for all of the codes considered for study. Also, a few graphs are plotted to observe the change in Bit Error Rate (BER) and probability with respect to changing code rates. The chapters of the thesis are divided into seven chapters as highlighted below.

The **first** chapter consists of the introduction of coding theory and its relevance in our day-to-day lives. The beginning of codes used in the form of integers and the required preliminaries for our study are mentioned.

The **second** chapter consists of integer codes correcting asymmetric CT-bursts and their probability of erroneous decoding. Encoding and decoding of the codes inflicted with asymmetric CT-burst errors are discussed. We give some comparisons with the similar kind of codes. The obtained results show that for many data lengths, the presented codes require less memory than their linear counterparts. The probability of erroneous decoding and BER of the codes are also presented. Further, we discuss how much such type of error may go undetected. The presented codes have the potential to be used in various practical systems, such as optical networks and VLSI memories.

The **third** chapter consists of the low-density and high-density asymmetric CT-burst correcting integer codes. These classes of integer codes are obtained by putting restriction on the weight of the asymmetric CT-burst discussed in Chapter 2. By doing so, the memory consumption and the code rates of the class get better. The probability of erroneous decoding and the BER are analysed by considering ϵ as the crossover probability in the Z-channel. Finally, an approach for identifying the errors that may go undetected in both the classes is presented.

The **fourth** chapter consists of the unidirectional solid burst correcting integer codes defined over the binary symmetric channel. The codes defined are very rate-efficient and have the capability of correcting the mentioned types of bursts up to their maximum possible length. Here, the probability of erroneous decoding is determined by considering the probabilities of $1 \to 0$ and $0 \to 1$ as equally likely.

The **fifth** chapter consists of the integer codes capable of correcting asymmetric solid burst errors. Unlike the codes presented in the preceding chapters, this code is capable of correcting the mentioned burst errors within a b-bit byte as well as between two adjacent b-bit bytes. By doing so, the code becomes capable of correcting the errors without interleaving. The probability of erroneous decoding is determined and analysed similarly to the preceding chapters, and finally, some properties of undetected errors are discussed.

The **sixth** chapter consists of the integer codes correcting asymmetric bursts occurring anywhere in the codeword. This class is a generalisation of the class of integer codes discussed in Chapter 5. Obviously, this class has a lower code rate compared to the preceding one. But in return, it provides error-correcting ability for all types of asymmetric burst errors. The probability of erroneous decoding is determined and a few graphs are plotted.

The **seventh** chapter of the thesis consists of the probabilities for existing integer codes correcting different types of symmetric and asymmetric errors. Errors here vary from code to code according to their occurrence (within a b-bit or between two b-bit bytes).

Throughout our study, we have tried to construct the codes in a better way based on the existing codes, correcting similar types of errors. In each of the codes, an attempt is made to make the code rate efficient and ensure that it consumes the least possible memory during the error-correcting procedure. We end our thesis with the required Python search codes used for finding the parity check matrix followed by the bibliographical references.