ASSIGNMENT 1

ELL710: CODING THEORY

Reed-Solomon encoding, decoding and error correction

SUBMITTED TO:
Prof. Manav Bhatanagar

Submitted by:
Anil Kumar (2023JTM2567)

Objective :-

Designing an RS(Reed-Solomon) (15, 7) code, where the message length is 7 symbols and the encoded codeword length is 15 symbols, operating over the finite field (GF (2^4)). You will implement an RS encoder in MATLAB to encode a random message of 7 symbols and transmit it over a simulated noisy channel that randomly corrupts up to 3 symbols in the codeword. Subsequently, you will implement an RS decoder to correct the errors and retrieve the original message. Through this, you will investigate the error correction capabilities of the RS (15, 7) code and analyze its performance by simulating different error scenarios. The goal is to determine how many errors the code can successfully correct and explain the limitations of Reed-Solomon codes in terms of error tolerance. You will submit the MATLAB code for encoding, error introduction, and decoding, along with a brief analysis of the results, including a comparison between the original, corrupted, and corrected messages.

Steps to perform the Reed Solomon coding-

1. Constructing Galois Field (GF).
2. Generates Reed-Solomon Generator Polynomial.
3. Encodes the Message.
4. Introduce Errors.
5. Decoding Process (Using Berlekamp Algorithm) –
    i. Calculate syndrome vector from the corrupted codeword.
    ii. Compute the error locator polynomial using syndrome values.
    iii. Identify error locations by determining roots of the error locator polynomial.
    iv. Calculate error magnitudes at identified locations.
    v. Adjust the corrupted codeword based on error locations and magnitudes to recover the original message.
6. Verification of Decoding –
    i. Check for detected errors based on the syndrome vector.
    ii. If errors are detected and corrected, display the recovered data.
    iii. Compare recovered data with the original message to confirm successful decoding.
    iv. If there is error in decoding then shows a message that some error has occurred.

Functions used-

1. **Main_function-** It took number of errors as input from user and then shows all the steps along with the final result.
2. **main_plotting-** This function varies number of errors from 1 to 9 and then for 1000 iterations. It find the correct decoding probability along with each number of errors and plot the graph.
3. **gf_div –** Used to divide two galois fields.
4. **gf_mul -**Used to multiply two galois fields.

5. **poly_diff-** Used to find derivative of polynomial .
6. **poly_div –** Used to divide to polynomials.
7. **poly_mul-** Used to multiply to polynomials.
8. **poly_sum -** Used to add to polynomials in modulo 2.
9. **rs_decode -** This function decodes received data.

Results-

Result of main_function:

When number of symbol errors is 3 it decoded correctly.

```
Give number of errors (Max 4 errors can be corrected by this code) :
3
```

```
Constructing GF(2^4) powers table..
Prime polynomial:
     1     1     0     0     1

Prime element: 2
Galois Field constructed successfully
Calculating Reed Solemon generator polynomial
Prime used to construct G: 2, Dmin: 9
Constructed code N=15 K=7 R=8 over GF(16)
G(x):     12    14     6    13     4     3     4     9     1
```

```
------------------
 ENCODING...
------------------
Size of message symbols (Random) to encode(7 symbols): Message symbols are:    12    3    7    14    15    2    4

Encoded codeword (15 symbols):
    4    4    14    6    13    7    8    10    4    2    15    14    7    3    12
```

```
------------------------------------------------
 CORRUPT WITH ERROR...
------------------------------------------------------------
Randomized error vector:
     0     5     0     0     0     0     0     0     0     0     7     0    15     0     0

Real error count: 3
Corrupted codeword:
     4     1    14     6    13     7     8    10     4     2     8    14     8     3    12
```

```
-------------------------------------
 DECODEING....
-----------------------------------------------------
Berlekamp gives us locator polynomial of power 3
L(x):
     1    10     8     5

Omega(x):
     6     3    12     0     0     0     0     0

ILocator 6 has error value 7 and pos 10
ILocator 8 has error value 15 and pos 12
ILocator 9 has error value 5 and pos 1
Syndrome vector:     6    10     1     8     3     4    11    11

Decoder detected error presence
Decoder thinks that we have correct data..
Recovered data:
    12     3     7    14    15     2     4

Errors in recovered data:
     0     0     0     0     0     0     0

-------------------------
 DECODING SUCCESSFUL
-------------------------
```

When number of errors is 5 decoder does not decoded correctly and it shows that "decoder detected error presence and can't recover data":

```
Give number of errors (Max 4 errors can be corrected by this code) :
5
Constructing GF(2^4) powers table..
Prime polynomial:
     1     1     0     0     1

Prime element: 2
Galois Field constructed successfully
Calculating Reed Solemon generator polynomial
Prime used to construct G: 2, Dmin: 9
Constructed code N=15 K=7 R=8 over GF(16)
G(x):     12    14     6    13     4     3     4     9     1
```

```
--------------------
 ENCODING...
--------------------
Size of message symbols (Random) to encode(7 symbols): Message symbols are:     14     3    11     1    12     3    11

Encoded codeword (15 symbols):
     7    11     1     1     4     8     7     2    11     3    12     1    11     3    14

-----------------------------------------------
 CORRUPT WITH ERROR...
-----------------------------------------------------------
Randomized error vector:
     0     0     3     9     0     0     0     0     0     6     0     0    13    10     0

Real error count: 5
Corrupted codeword:
     7    11     2     8     4     8     7     2    11     5    12     1     6     9    14
```

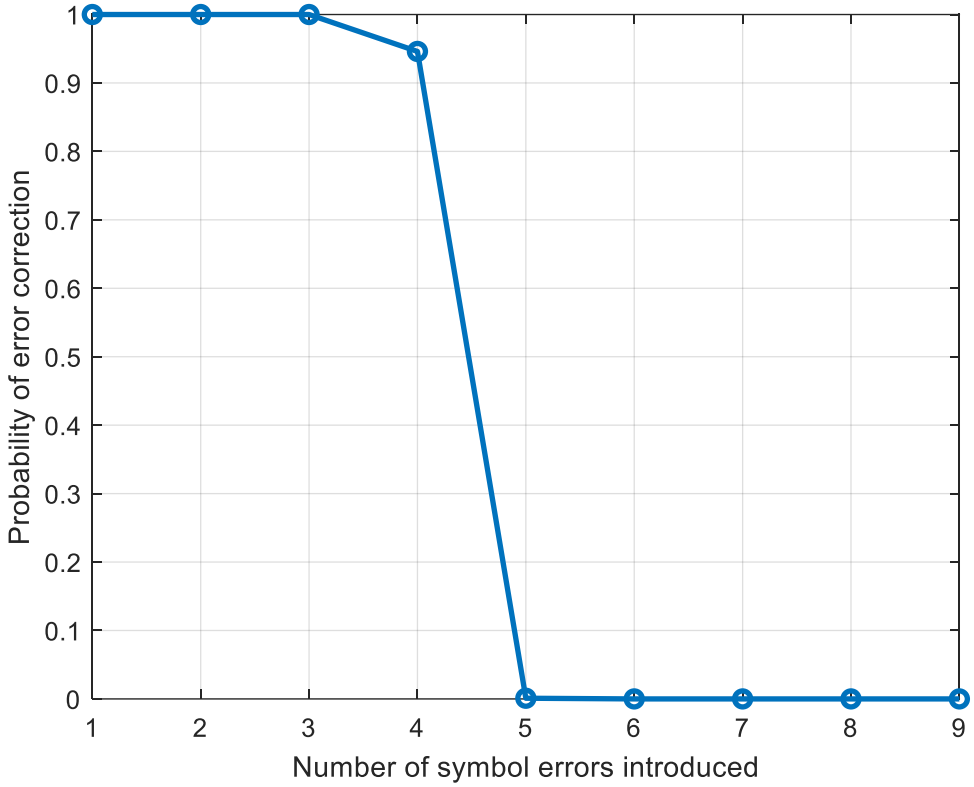
```
--------------------------------------
 DECODEING....
-----------------------------------------------------------
Berlekamp gives us locator polynomial of power 4
L(x):
     1     10     14      7      0

deg L(x) < V. Unable to correct errors
Syndrome vector:    13    11    13    15     2     7     7     1

Decoder detected error presence
Decoder cannot recover data
```

Result of main_plotting function:

I have varied the number of symbol errors from 1 to 9 and plotted the probability that the decoder can successfully correct the symbol errors versus the number of symbol errors. The plot shows that the RS(15,7) code is capable of correcting up to 3 symbol errors consistently. For 4 symbol errors, the decoder occasionally manages to correct them, but not always. However, when the number of symbol errors exceeds 4, the decoder fails to correct the code in every trial, even after 1000 iterations.



Reed-Solomon codes offer robust error correction, particularly for burst and random errors, but they have clear limits in error tolerance. These limits are defined by the code parameters (n,k), and once the number of errors exceeds the code's designed capacity, the decoder becomes unreliable. Additionally, burst errors, computational complexity, and trade-offs between data throughput and error correction are important considerations when using RS codes.

In general, in each block of n digits an (n, k) RS code can:

- correct up to (n-k)/2 symbol errors
- correct up to n − k erasures.