Programming Assignment 1 (RS Codes and List Decoding of RS Codes)

Prasad Krishnan
prasad.krishnan@iiit.ac.in

Instructions

- This assignment is created in a graded manner so as to aid you in writing the code for encoding and decoding of RS Codes. Hence proceed in the order as given in the assignment, solving each problem completely. Some questions (specifically the first) require you to find out the answers in a elaborate manner and check them with the inbuilt algebra functions or toolboxes of your favourite programming languages. This is to enable you to not depend on these high-level toolboxes and implement some basic algebra by yourself.
- Please use Python (whatever extension of Python is applicable) or Matlab.
- Your code should be clean and should carry helpful comments at every stage (please include comments about any special functions you use as it is useful for the reader).
- Please add helpful English statements when the code runs to enable the user to understand what (s)he is giving as input and how it should be given.
- Submit both the code and the outputs (make sure that the outputs are presented 'stage by stage' with suitable textual commentary to the reader, as though you are testing the code at each stage, and not just the final output is provided).
- Marks (5 Questions): 5+5+10+15+15 (even if the program runs correctly, depending on the instructions followed the marks will be modulated)

Assignment

1. (a) Find an irreducible polynomial of degree 8 over $\mathbb{F}_2$.

   (b) In the finite field $\mathbb{F}_2$ defined using the above irreducible polynomial, find a primitive element (call it β) and show it as a polynomial.

Find the product of β^{34} and β^{20} (find product by expressing them as polynomials and doing the modulo algebra explicitly). Calculate the polynomial representation of β^{54} in the same field. Check if these two answers are the same.

(c) In the same way above practice constructing finite fields of various sizes $\mathbb{F}_{2^m}$ $(m = 1, 2, 3, ..)$ and make sure you get the algebra. [Note:This question is not part of the submission]

2. Calculate and display the encoded vectors (codewords) corresponding to the following RS codes and message polynomials (use characteristic 2 finite fields, i.e., of the form $\mathbb{F}_{2^m}$ only). [Note : β is a primitive element in each field]

   (a) $[n = 10, k = 4]$ code, message polynomial $M(X) = 1 + \beta^2 X$.

   (b) $[n = 20, k = 10]$ code, message polynomial $M(X) = \beta^4 + \beta^5 X^3 + \beta^{10} X^9$.

   (c) $[n = 50, k = 25]$ code, message polynomial $M(X) = \beta^{20} + \beta^4 X^8 + \beta^{44} X^{12} + \beta^{30} X^{14} + \beta^{15} X^{18} + \beta^3 X^{23}$.

3. Take each of the codes above in Q2. Allow the user to specify a message polynomial by specifying the non-zero monomials and their coefficient values. In this part you will allow user to introduce upto $\lfloor \frac{d_{min}(\mathcal{C})-1}{2} \rfloor$ errors in the codewords and calculate received codeword. Allow the user to input error monomials and coefficient values as user input in your code and also output the message vector (or polynomial), the error vector and the received vector. Code the unique decoding algorithm and using that show that the codeword is decoded correctly. Please show all steps in the output.

4. Recognize that in the List Decoding Algorithms, the degree of Y denotes the max size of the list that can be formed (because in the factorization/root-finding step, $(Y - \hat{M}(X))$ should divide $Q(X, Y)$). Use this idea along to play around with the degree of Y (start with $deg_Y(Q) = 2$) and appropriate degree for X to implement Algorithm 1 and correct **at least one more** than $\lfloor \frac{d_{min}(\mathcal{C})-1}{2} \rfloor$ errors **with a list of size** > 1 (again you have to enable the user to input some message polynomial and error polynomial with more than $\lfloor \frac{d_{min}(\mathcal{C})-1}{2} \rfloor$ errors. Display a complete example as the output with all the intermediate steps).

   - Note 1: Multivariate factorization, which was not only assumed and not described in the class, is thankfully available in both Matlab and SciPy. Search online and figure out.
   - Note 2: In case you have memory overflow or the time involved in the computations seem very high, then please assume a code with same length but lower dimension, and try. Specify such changes in your code/output.

5. Follow similar observations and instructions as Q4 above and code Algorithm 2 to correct more than $\lfloor \frac{d_{min}(\mathcal{C})-1}{2} \rfloor$ errors. Same notes applies as in Q4.