Report of LDPC Codec based on C

- **Introduction of Algorithms**

  In this project， the system of LDPC codec divided four parts：

  1. Construct the parity-check matrix H and encode.
  2. Passing the awgn channel after using BPSK modulation.
  3. Decoding with normalize min-sum algorithm.
  4. Compute the BER (bit error rate) and FER (frame error rate).

In the first part, we given a matrix (dimension of 46x68 smaller than the parity-check matrix we really use in encoding), So we should treat every elements in this matrix as a 32x32 matrix, if this element is 0， means to this element should replace to a Identity matrix, similarity, the element -1 is replaced by zeros matrix. But, if the element is an integer, we can treat it as a indentity matrix whith right shift (the number of shift is depended on the integer).

There are two way to construct LDPC code,one is by using the Gaussian Elimination to get the parity-check matrix in systematic form and then construct the generator matrix, the other way is focusing on the structure of 5G H matrix where has dual diagonal, therefore we can find P1 by knowing message bits. In detail, we can get a equation of the each row, and solve P1 P2 ...in turn.Due to the Gaussian Elimination is hard to implement and ensure the correctness of the elementary transformation, so this project I choice the second way to generate the parity bits and finish encode.

The second part is BPSK modulation, by simple way, mapping 1 to -1 and 0 to 1,and the processed signal transition to the AWGN channel in term of plus the AWGN noise.

The third part is LDPC decoder algorithm ,this project use the normalize min-sum algorithm which is the approximation of SPA.

We can initialize the L_j for the channel mode, because we use the AWGN channel ,so we use $L_j = 2y_j \big/ \sigma^2$ (it is worth mentioning that y_j can be directly used in this algorithm to approximate),and if h_{ij}=1,we think there is a edge between the CN_i and VN_j , and there has information exchange between them.For this reason, we can initialize L_{ji} by L_j. After initialization we can update the CN by

$$L_{i\to j} = \prod_{j'\in N(i)-\{j\}} \alpha_{j'i} c_{atten} \cdot \min_{j'\in N(i)-\{j\}} \beta_{j'i}$$, and this project set C_{atten} = 0.75.

and after VN update by

$$L_{j\to i} = L_j + \sum_{i'\in N(j)-\{i\}} L_{i'\to j}$$

After above we compute the total LLR by

$$L_{\mathrm{j}}^{total} = L_j + \sum_{i\in N(j)} L_{i\to j}$$

Update the information of the two nodes according to the set maximum number of iteration.After the iteration, we do a decision about the estimate bits. If the total LLR<0 , we think

the transmit bit is 1, otherwise we think the transmit bit is 0.

At the beginning , this project using the case with flooding scheduling, but find it's performance isn't good as image, so to inprove the decode method by using laryer scheduing. As result, the case with laryer scheduing has faster convergence.

The calculation of BER and FER is performed at the same time as the simulation. When the error threshold of the frame is exceeded, the cyclic is break.

- **Simulation results**

This program sets format IO, to run the program ,we should key in

ldpc.exe 'number of SNR steps' 'interation' 'error threshold' 'attenuation'

Where,the step length of SNR is 0.25 (it means that if we key in 5,we can get the SNR cases of [1 1.25 1.5 1.75 2]. 'interation'is the maximum number of interaction, 'error threshold' decide how many errors will beak cyclic,It also affects the time required by the program. 'attenuation' is the normalize coefficient to avoid too optimistic decide.

We can select these parameters for simulation as needed. The following are the simulation results of the two cases tested.

Case I: Input "5 7 10 0.75"
SNR=1.0:0.25:2.0 / Iteration:7 / Error threshold:10 / Attenuation:0.75

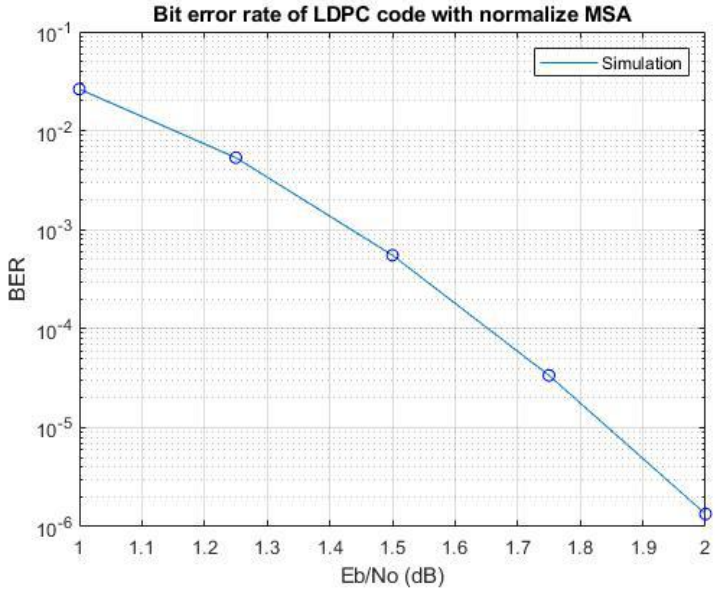
SNR from 1.0 to 2.0	iteration : 7	Error threshold : 10		Attenuation : 0.75	
SNR	Frame_error	Total_frame	Bit_error	BER	FER
1.00	10	19	478	3.57 E-002	5.26 E-001
1.25	10	39	199	7.25 E-003	2.56 E-001
1.50	10	101	83	1.17 E-003	9.90 E-002
1.75	10	2048	99	6.87 E-005	4.88 E-003
2.00	10	10601	32	4.29 E-006	9.43 E-004

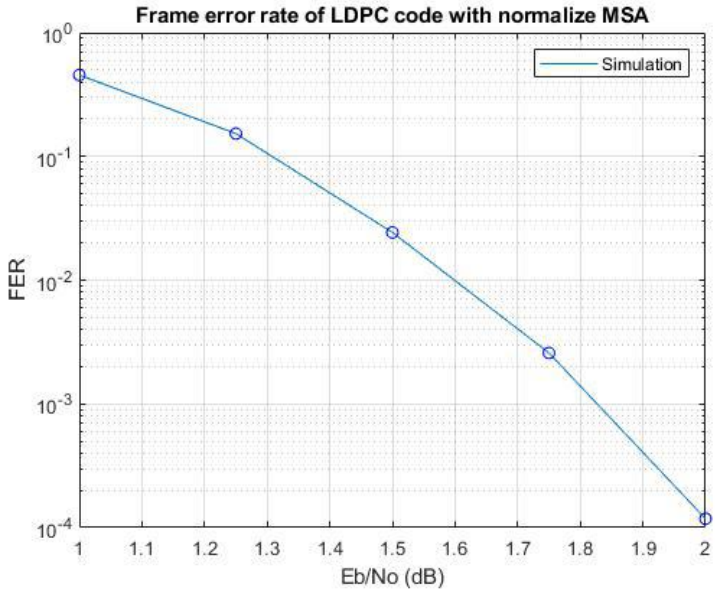
We can find the performance of "error threshold = 10" is not good, because too few frames can be tolerated, the test data is too small, the fluctuation is large, and the probability of poor results is greater. So we improve the "error threshold" to 100, and improve the "iteration number" to 8.

Case II: Input "5 8 100 0.75"
SNR=1.0:0.25:2.0 / Iteration:8 / Error threshold:100 / Attenuation:0.75

SNR from 1.0 to 2.0	iteration : 8	Error threshold : 100		Attenuation : 0.75	
SNR	Frame_error	Total_frame	Bit_error	BER	FER
1.00	100	220	4071	2.63 E-002	4.55 E-001
1.25	100	655	2462	5.34 E-003	1.53 E-001
1.50	100	4115	1599	5.52 E-004	2.43 E-002
1.75	100	38704	917	3.37 E-005	2.58 E-003
2.00	100	849234	802	1.34 E-006	1.18 E-004

We can draw BER (Bit error rate) and FER (Frame error rate) based on the data of the second case:





We can find from two picture, BER and FER decreasing as SNR increase. And each SNR point is very close to the reference data