NB LDPC and GLDPC Codes

for Future Communication Standards

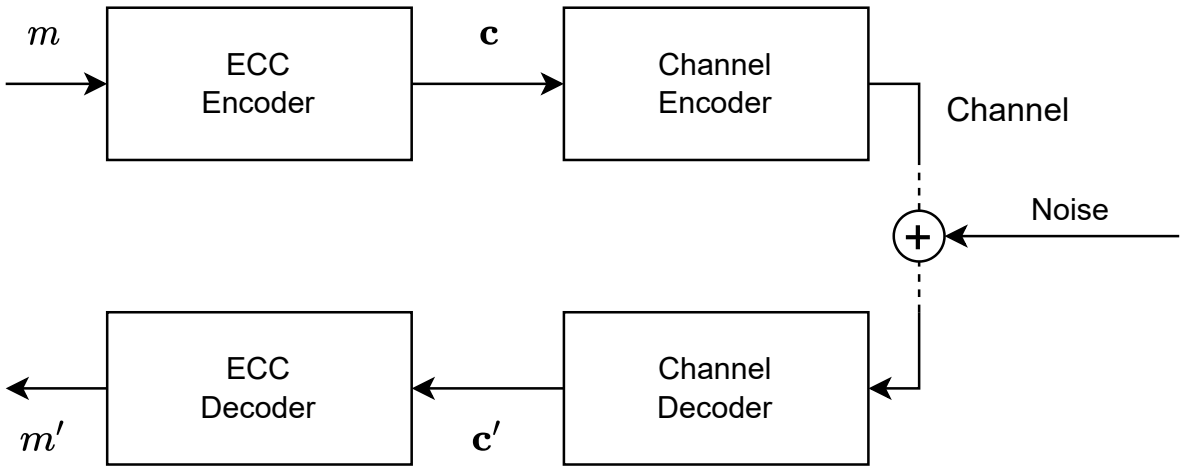
Sander Mikelsaar

May 10, 2022

Joint work with Irina E. Bocharova & Boris D. Kudryashov

Error-Correcting Codes

- Any communication over a noisy channel requires error-correcting codes (ECCs)
- We would like the sent message m to be the same as the received message m'



ECC Example

- Simplest example of an ECC is a repetition code
- For input $x \in \{0, 1\}$, encode it as $y \in \{000, 111\}$
- Allows for correction of 1 error
- Resulting code rate $R = 1/3$
- Redundancy is used to correct errors
- Balance redundancy and error rates

Channel Encoding (Modulation)

- Phase-shift keying (PSK) modulates the phase of a constant frequency carrier wave
- M-PSK maps length-$\log_2 M$ bitstrings to M channel symbols
- Simplest is BPSK (or 2-PSK):
  For input $c \in \{0, 1\}$, map it to $v = \sqrt{E_s}(2c - 1)$, where E_s is the carrier signal energy

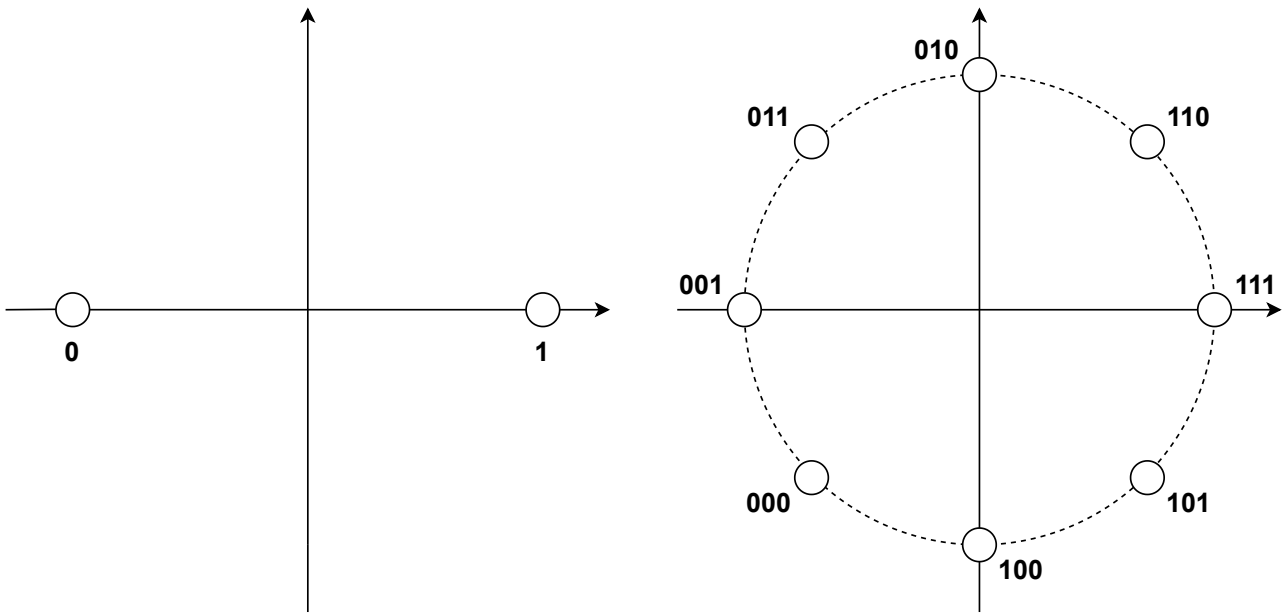
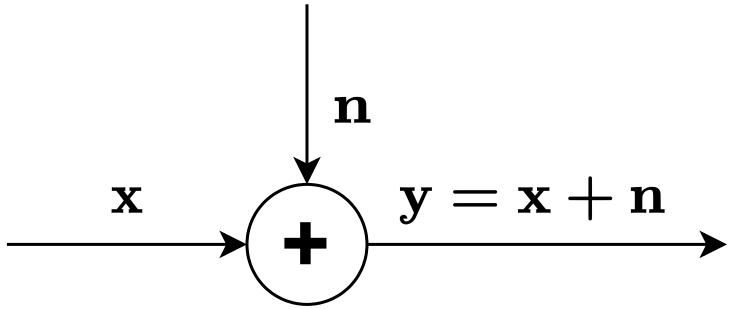


Figure: BPSK and 8-PSK (with Gray code mapping) signal constellations

AWGN

- Additive White Gaussian Noise (AWGN) channel
- Typical channel model for simulations
- Noise n is a vector of independent zero-mean Gaussian random variables with variance $\sigma^2 = N_o/2$
- The signal-to-noise ratio $SNR = 10 \log_{10} \frac{E_s}{N_o}$ (dB)



Codes

- A generator matrix G of a linear $[n, k]$ code $\mathcal{C}$ is an $k \times n$ matrix whose rows are basis vectors.
- $G = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$ for the $[3, 1]$ repetition code
- For a message $\mathbf{m} = (m_1, ..., m_k)$, the corresponding codeword $\mathbf{c} = (c_1, ..., c_n) = mG$
- A parity-check matrix H for the same code is a $r \times n$ matrix, where $r = n - k$ and $GH^T = \mathbf{0}$ and $\forall \mathbf{c} = \mathbf{m}G : \mathbf{c}H^T = \mathbf{0}$
- Rate R of a code is $R = \frac{k}{n}$

LDPC codes

- Low density parity-check (LDPC) codes
- Very sparse parity-check matrix H
- In practice, long LDPC codes are needed
- DVB-S2 standard, $n = 64800$

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

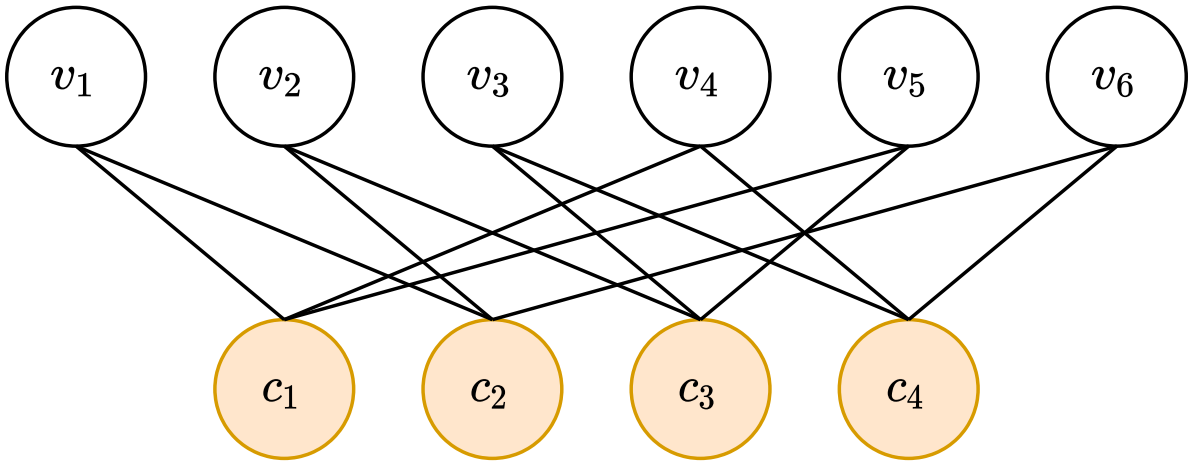
Tanner Graph I

- Tanner graph is a bipartite graph
- Can be used to represent matrix H
- Denote columns as variable nodes and rows as check nodes
- A symbol "1" in H represents an edge in the Tanner graph

$$H = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Tanner Graph II

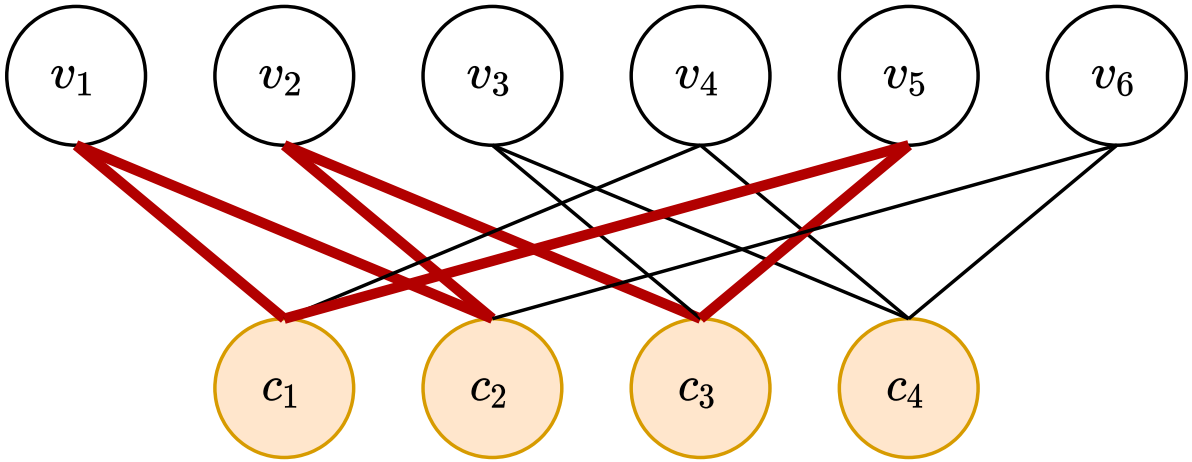
$$H = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$



Tanner Graph Cycles I

- A cycle (circuit) is a path in a graph that starts and ends at the same node
- Small Tanner graph cycles should be avoided

$$\mathbf{c} = (c_1 \leftarrow v_1 \rightarrow c_2 \leftarrow v_2 \rightarrow c_3 \leftarrow v_5 \rightarrow c_1)$$



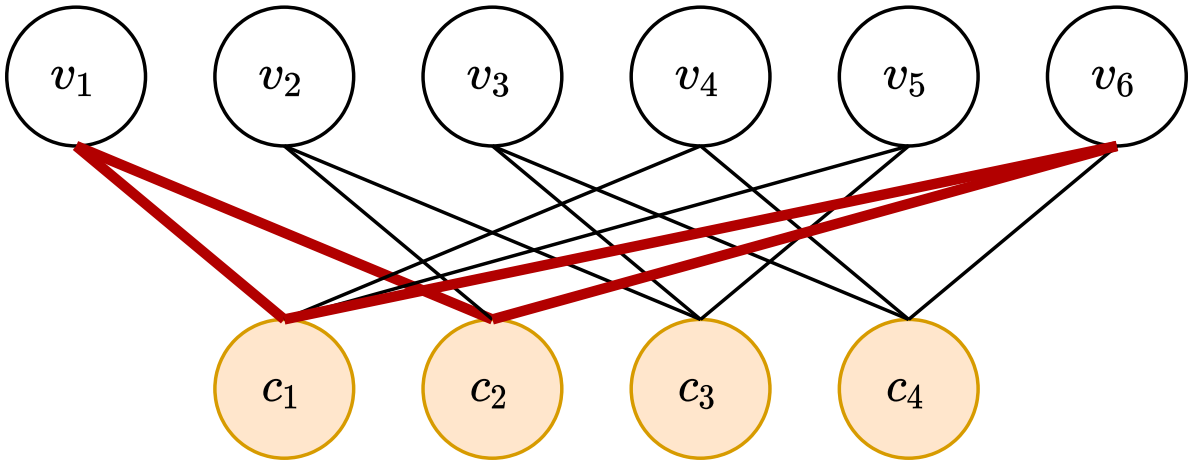
Tanner Graph Cycles II

- Girth g of a Tanner graph is the size of its smallest cycle
- $g = 6$ for the Tanner graph of H shown below
- Cannot add an edge without reducing g

$$H = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Tanner Graph Cycles III

$$H = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$



$$\mathbf{c} = (c_1 \leftarrow v_1 \rightarrow c_2 \leftarrow v_6 \rightarrow c_1)$$

QC LDPC I

- We can use the previous matrix H as a *base matrix* H_b
- Replace "1"s in H_b by $M \times M$ cyclic permutation matrices And "0"s by $M \times M$ all-zero matrices
- For $M = 3$, we can label "1"s in H_b as d^i for $i \in \{0, 1, 2\}$:

$$d^0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad d^1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad d^2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

- From a $r \times n$ base matrix H_b, we get $rM \times nM$ lifted matrix H

QC LDPC II

- We can store large LDPC code matrices using *degree matrices*

$$H_d = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} d^0 & 0 & 0 & d^0 & d^0 & d^1 \\ d^0 & d^0 & 0 & 0 & 0 & d^0 \\ 0 & d^0 & d^0 & 0 & d^0 & 0 \\ 0 & 0 & d^0 & d^0 & 0 & d^0 \end{pmatrix} \end{array}$$

- A cycle in the base matrix H_b exists in the lifted matrix H if the alternating sum of edge degrees *mod* M equals 0
- Signs correspond to directions of arrows:

$$\mathbf{c} = (c_1 \overset{0}{\leftarrow} v_1 \overset{0}{\rightarrow} c_2 \overset{0}{\leftarrow} v_6 \overset{1}{\rightarrow} c_1) \rightarrow (-0 + 0 - 0 + 1) \mod 3 = 1$$

Decoding

- Optimal decoding is not practical for LDPC codes
- Sub-optimal techniques with linear complexity are required
- Use "soft input soft output" (SISO) algorithms
- Iterative decoding

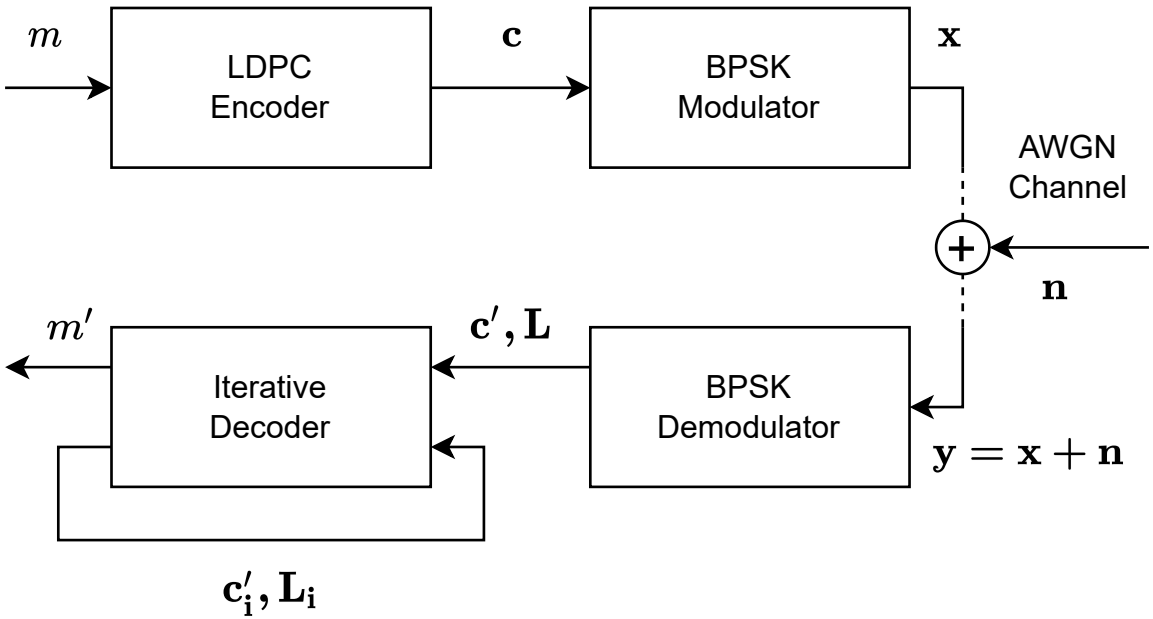
BP Decoding

- Belief propagation (BP) decoding
- Calculate probabilities symbol x_i to be either 0 or 1 based on the received sequence $\mathbf{y}$
- $p_i = Pr[x_i = c|\mathbf{y}]$, where $c \in \{0, 1\}$
- Let event S_j denote satisfied check in position j
  Then $p_{j,i} = Pr[x_i = c|\mathbf{y}, S_j]$
- Let event S denote that all checks containing x_i are satisfied

$$\frac{Pr[x_i = 0|\mathbf{y}, S]}{Pr[x_i = 1|\mathbf{y}, S]} = \frac{(1 - p_i)}{p_i} \prod_{j=1}^{J} \frac{1 + \prod_{h=1, h\neq i}^{K}(1 - 2p_{j,h})}{1 - \prod_{h=1, h\neq i}^{K}(1 - 2p_{j,h})}$$

LDPC Decoding I

- Decoding of LDPC codes is performed iteratively
- Decode until maximum number of iterations is reached or zero-syndrome is achieved
- For code length n, each decoding iteration has linear complexity $\mathcal{O}(n)$



LDPC Decoding II

- Cycles of the Tanner graph impact decoding performance
- Iterative decoding *error floor* phenomenon caused by *trapping sets*
- Avoiding small cycles significantly improves decoding performance
- Girth and number of smallest cycles is indicative of code performance

NB LDPC I

- Binary LDPC codes are used in modern standards
- Digital TV (DVB-S2), Gigabit networking (ITU-T G.hn, 10GBASE-T Ethernet), WiFi (802.11n and 802.11ac), 5G (3GPP 5G NR)
- Non-binary (NB) LDPC codes achieve better BP decoding performance

NB LDPC II

- Generalization of binary LDPC codes to $GF(q)$, $q = 2^m, m > 1$
- Substitute non-zero elements of H by elements of $GF(q)$
- Decoding is done over $GF(q)$
- For a $r \times n$ matrix H of a NB LDPC code over $GF(2^m)$, a binary image is obtained by replacing non-zero elements by $m \times m$ companion matrices
- For a NB LDPC code of length n over $GF(2^m)$, transmit a binary image of the codeword $\mathbf{c}$

$$\mathbf{c} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_n\}, \text{ where } \mathbf{c}_i = \{c_{i,1}, c_{i,2}, ..., c_{i,m}\}$$

GLDPC I

- Generalized LDPC (GLDPC) codes are another powerful generalization of binary LDPC codes
- Constructed using a base matrix H_b and a constituent code defined by a parity-check matrix H_c
- For $r_b \times n$ matrix H_b with uniform row weight K, use $r_c \times K$ matrix H_c
- Resulting $r_c r_b \times n$ binary H of the GLDPC code

GLDPC II

- Non-zero elements of H_b replaced by columns of H_c
- Zero elements replaced by all-zero columns of length r_c
- Choose columns such that the number of short cycles is minimized

$$H_b = \begin{pmatrix} c_1 & 0 & 0 & c_2 & c_3 & 0 \\ c_2 & c_1 & 0 & 0 & 0 & c_3 \\ 0 & c_3 & c_2 & 0 & c_1 & 0 \\ 0 & 0 & c_3 & c_1 & 0 & c_2 \end{pmatrix} \quad \mathrm{H}_c = \begin{array}{c} \begin{matrix} c_1 & c_2 & c_3 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

GLDPC Code Construction

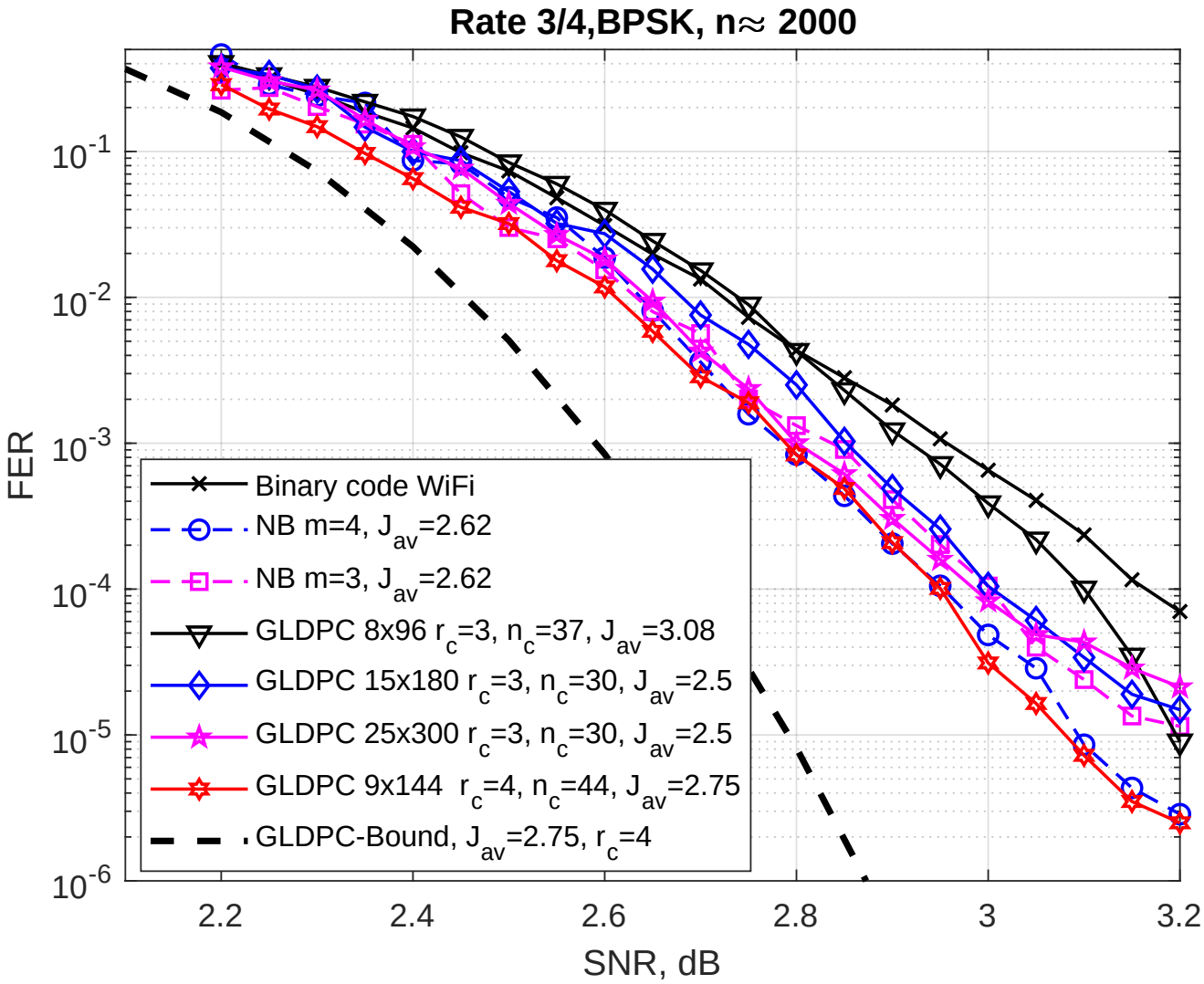
- Construct row-regular base parity check matrices H_b of varying sizes and average column weights
- Construct binary QC LDPC codes using lifting factor M
- Replace non-zero elements by columns of constituent code H_c to obtain QC-GLDPC codes
- Maximize girth and minimize amount of smallest cycles

H_b	M	QC-LDPC	H_c	GLDPC
8×96	21	168×2016	3×37	504×2016
15×180	11	165×1980	3×30	495×1980
25×300	7	175×2100	3×30	525×2100
9×144	14	126×2016	4×44	504×2016

Results I

- To compare performance, NB LDPC and GLDPC are simulated over AWGN channel using BPSK
- Frame error rate (FER) is measured for BP decoding
- Max 50 decoding iterations
- Simulate until 25 frame errors encountered
- $R = 3/4$ GLDPC codes of length $n \approx 2000$
- $R = 3/4$ NB QC LDPC codes over $GF(2^3)$ and $GF(2^4)$
- Compare with binary codes used for WiFi

Results II


Rate 3/4,BPSK, n≈ 2000
FER
SNR, dB
10^-1
10^-2
10^-3
10^-4
10^-5
10^-6
2.2
2.4
2.6
2.8
3
3.2
Binary code WiFi
NB m=4, J_av=2.62
NB m=3, J_av=2.62
GLDPC 8x96 r_c=3, n_c=37, J_av=3.08
GLDPC 15x180 r_c=3, n_c=30, J_av=2.5
GLDPC 25x300 r_c=3, n_c=30, J_av=2.5
GLDPC 9x144 r_c=4, n_c=44, J_av=2.75
GLDPC-Bound, J_av=2.75, r_c=4

Conclusions

- Both NB LDPC and GLDPC codes outperform their binary counterparts
- GLDPC codes offer slightly better performance
- Both classes of codes achieve relatively low decoding complexity

Thank you!