Low-Density Parity-Check (LDPC) codes

Binary Code vector $\boldsymbol{v} = (v_1, v_2, \cdots, v_N)$
BPSK modulation
Discrete memoryless AWGN channel, $\boldsymbol{e} = (e_1, e_2, \cdots, e_N)$
Received vector: $\boldsymbol{y} = (y_1, y_2, \cdots, y_N) = \boldsymbol{v} + \boldsymbol{e}$

Maximum a posteriori (MAP) detection:
To correctly detect the j-th element of $\boldsymbol{v}$, *a posteriori probability* (APP) $\Pr(v_j|\boldsymbol{y})$ is evaluated, and select the maximum one among $\Pr(v_j = 0|\boldsymbol{y})$ and $\Pr(v_j = 1|\boldsymbol{y})$.

In general, a longer linear block code (larger N for the same code rate R) can provide a better correction capability. Extremely long LDPC codes can be constructed to approach channel capacity.

But larger N also introduces higher complexity.
LDPC codes separate (simplify) the optimum MAP to suboptimum iterative **message passing** decoder.

Graphical representation for LDPC codes:

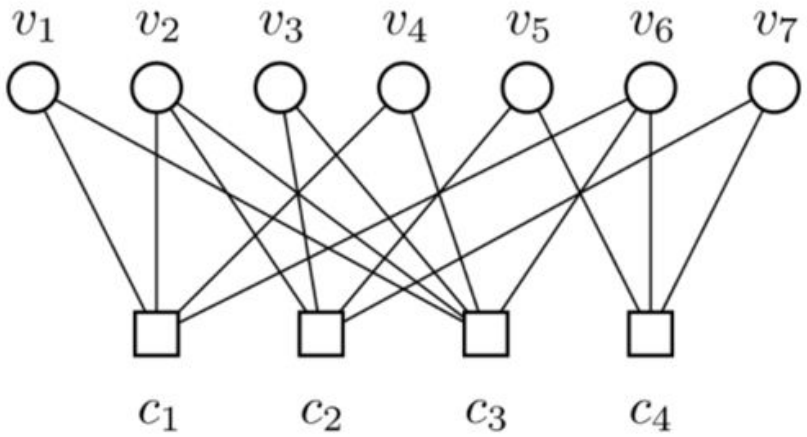
LDPC code is a linear block code:
Parity-Check Matrix $\mathbf{H}$ $\longleftrightarrow$ Tanner graph

A Tanner graph is a bipartite graph.
The nodes are separated into two types.
Edges connecting only nodes of different types.

variable nodes (code-bit nodes) : VN, or V
check nodes (constraint nodes) : CN, or C

C_i is connected to V_j whenever element h_{ij} in $\mathbf{H}$ is 1.

$$\mathbf{H} = \begin{array}{c} \begin{matrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{matrix}$$



Message Passing Decoding

Likelihood ratio, LR:

$$l(v_j|\boldsymbol{y}) \triangleq \frac{\Pr(v_j = 0|\boldsymbol{y})}{\Pr(v_j = 1|\boldsymbol{y})} = \frac{\Pr(\boldsymbol{y})}{\Pr(v_j = 1, \boldsymbol{y})} \frac{\Pr(v_j = 0, \boldsymbol{y})}{\Pr(\boldsymbol{y})}$$

$$= \frac{\Pr(v_j = 1)}{\Pr(v_j = 1, \boldsymbol{y})} \frac{\Pr(v_j = 0, \boldsymbol{y})}{\Pr(v_j = 0)} = \frac{\Pr(\boldsymbol{y}|v_j = 0)}{\Pr(\boldsymbol{y}|v_j = 1)}$$

Since the channel is discrete memoryless AWGN:

$$l(v_j|\boldsymbol{y}) = \frac{\Pr(\boldsymbol{y}|v_j = 0)}{\Pr(\boldsymbol{y}|v_j = 1)} = \prod_{i=1}^{N} \frac{\Pr(y_i|v_j = 0)}{\Pr(y_i|v_j = 1)}$$

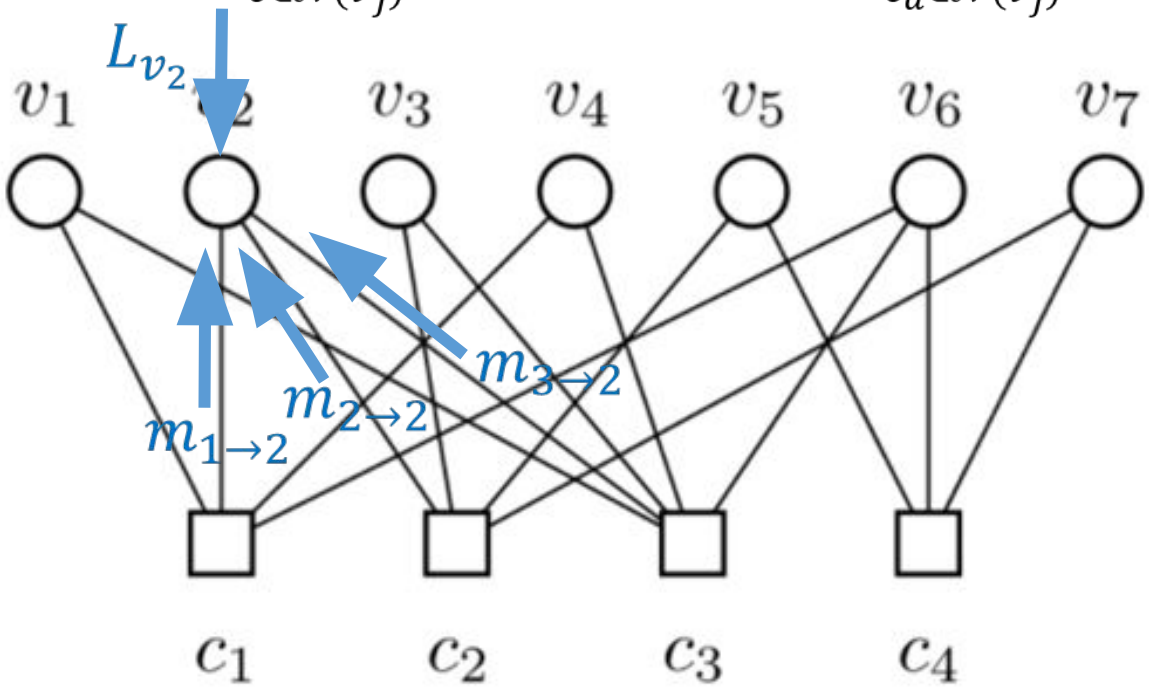
Consider the numerical issue, log-likelihood ratio is in favor:

$$L(v_j|\boldsymbol{y}) = log\left(\prod_{i=1}^{N} \frac{\Pr(y_i|v_j = 0)}{\Pr(y_i|v_j = 1)}\right) = \sum_{i=1}^{N} log\left(\frac{\Pr(y_i|v_j = 0)}{\Pr(y_i|v_j = 1)}\right) = \sum_{i=1}^{N} L(v_j|y_i)$$

The low-density nature of the $\mathbf{H}$ facilitates iterative decoding. The suboptimum reliability can be evaluated by the LLR of the sample y_i, and the messages from the check nodes neighboring to v_j:

$$L(v_j|\mathbf{y}) \cong log\left(\frac{\Pr(y_j|v_j=0)}{\Pr(y_j|v_j=1)}\frac{\Pr\left(\boldsymbol{m}_{c\to v,c\in\mathcal{N}(v_j)}\middle|v_j=0\right)}{\Pr\left(\boldsymbol{m}_{c\to v,c\in\mathcal{N}(v_j)}\middle|v_j=1\right)}\right)$$

$$= L(v_j|y_i) + \sum_{c\in\mathcal{N}(v_j)} L(v_j|\boldsymbol{m}_{c\to v}) = L_{v_j} + \sum_{c_a\in\mathcal{N}(v_j)} m_{a\to j} = L_j$$



Sum-Product Algorithm (one of the most popular message passing)**:**

Variable node operation: (generate V2C message)

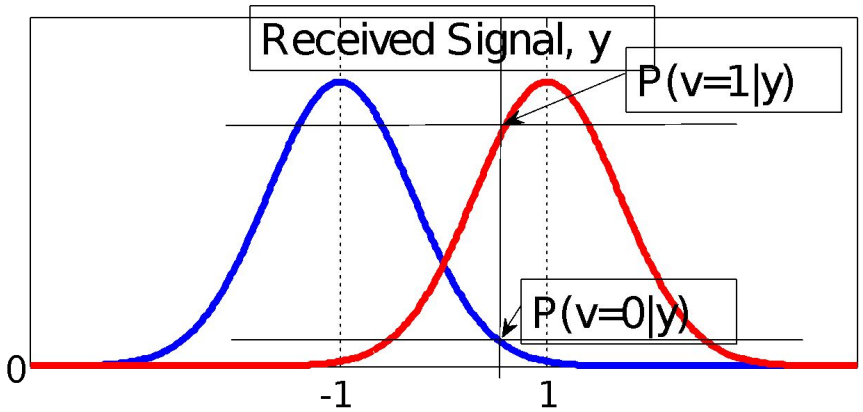
$$L_{j\to i} = L_{v_j} + \sum_{c_a \in \mathcal{N}(v_j)\backslash c_i} m_{a\to j}$$

Check node operation: (generate C2V message)

$$m_{i\to j} = 2\tanh^{-1}\left(\prod_{v_b \in \mathcal{N}(c_i)\backslash v_j} \tanh\left(\frac{L_{b\to i}}{2}\right)\right)$$

Decision:

$$L_j = L_{v_j} + \sum_{c_a \in \mathcal{N}(v_j)} m_{a\to j}$$



Example for low density H:

Since LDPC codes have extremely low density parity-check matrices, the complexity of message passing decoding keeps low.

The 'density' is defined as the ratio of '1's components in the parity-check matrix H.

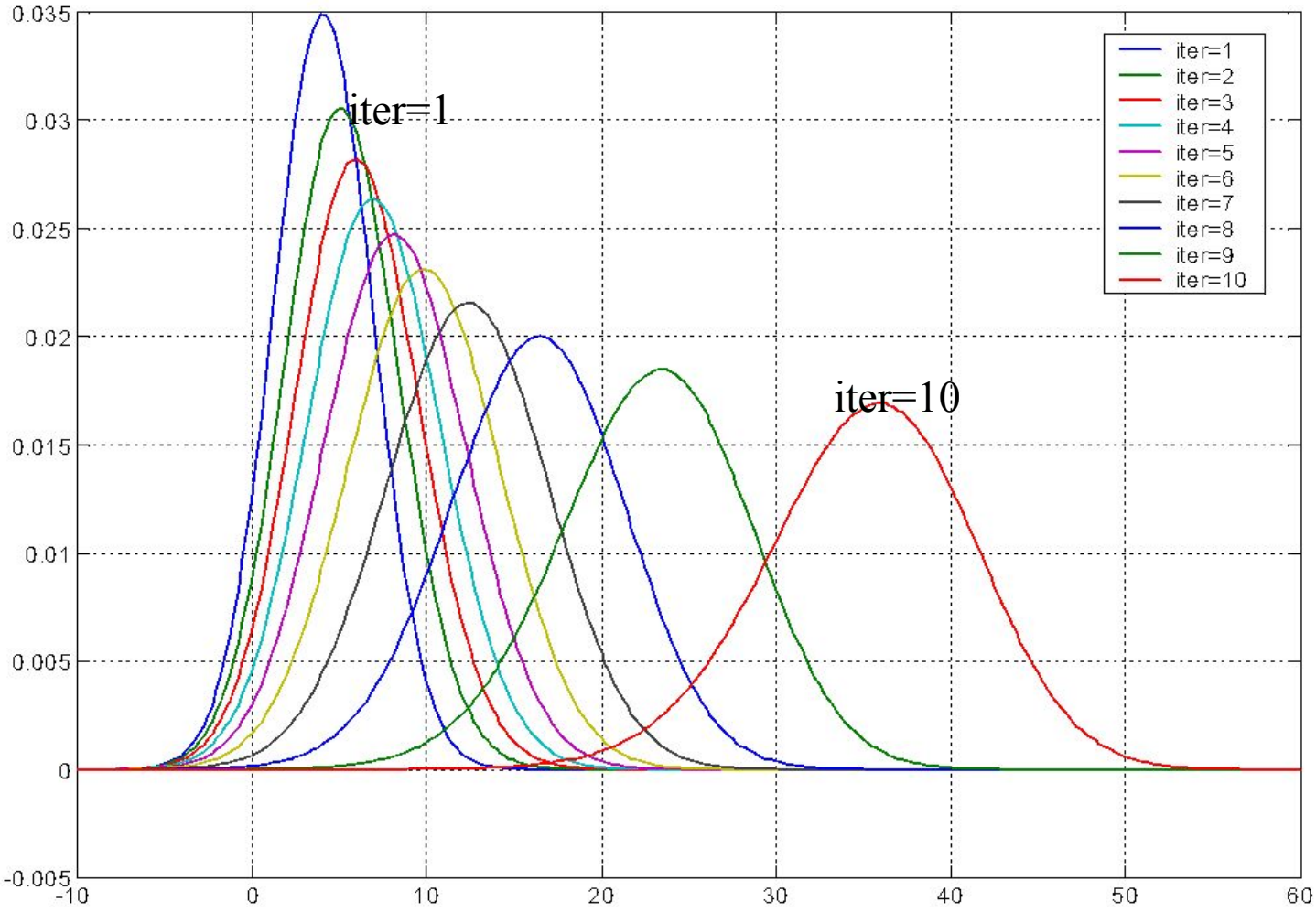
H for the (7, 4) Hamming code has a density of **13/28**

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

H for the (63, 51) BCH code has a density of **372/63/12=0.4921**

H for the (1944, 972) WiFi LDPC code has a density of **6966/1944/972=0.0037**

The distribution of variable-node values v.s. number of iteration:



Message Passing Decoding

The low-density nature of the $\mathbf{H}$ facilitates iterative decoding.

The various iterative decoding algorithms are suboptimum, but they often provide near-optimum performance.

Message-passing decoding refers to a collection of low-complexity decoders working in a distributed fashion to decode a received codeword in a concatenated coding scheme.

Message-passing decoding can also be called belief propagation decoding.

Sum-Product algorithm (SPA) and Mini-Sum algorithm (MSA) are two of the most popular decoding algorithm for LDPC codes.

Check node operation: (generate C2V message)

Sum-Product Algorithm:

$$m_{i\to j} = 2\tanh^{-1}\left(\prod_{v_b\in\mathcal{N}(c_i)\setminus v_j}\tanh\left(\frac{L_{b\to i}}{2}\right)\right)$$

Min-Sum Algorithm:

$$m_{i\to j} = \min_{v_b\in\mathcal{N}(c_i)\setminus v_j}(|L_{b\to i}|)\prod_{v_b\in\mathcal{N}(c_i)\setminus v_j}\operatorname{sgn}(L_{b\to i})$$

Message Passing Decoding for LDPC codes

Single parity-check code MAP decoder and APP

Consider a vector of d independent binary random variables $\boldsymbol{a} = (a_0, a_1, \cdots, a_{d-1})$. The probability of each element a_l be 0 and 1 are

$$\Pr(a_l = 0) = p_0^{(l)} \text{ and } \Pr(a_l = 1) = p_1^{(l)}$$

The probability that $\boldsymbol{a}$ contains an even number of 1s is

When d=1: $\Pr(even|d = 1) = \Pr(a_0 = 0)$

$$= p_0^{(0)} = 1 - p_1^{(0)} = \frac{1}{2} + \frac{1}{2}\left(1 - 2p_1^{(0)}\right)$$

Suppose d=k: $\Pr(even|d = k) = \frac{1}{2} + \frac{1}{2}\prod_{l=0}^{k-1}\left(1 - 2p_1^{(l)}\right)$

Then d=k+1:

$$\Pr(even|d = k+1) = \Pr(even|d = k)\, p_0^{(k)} + \Pr(odd|d = k)\, p_1^{(k)}$$

Message Passing Decoding for LDPC codes

Single parity-check code MAP decoder and APP

Consider a vector of d independent binary random variables $\boldsymbol{a} = (a_0, a_1, \cdots, a_{d-1})$. The probability of each element a_l be 0 and 1 are

$$\Pr(a_l = 0) = p_0^{(l)} \text{ and } \Pr(a_l = 1) = p_1^{(l)}$$

The probability that $\boldsymbol{a}$ contains an even number of 1s is

When d=1: $\Pr(even|d = 1) = \Pr(a_0 = 0)$

$$= p_0^{(0)} = 1 - p_1^{(0)} = \frac{1}{2} + \frac{1}{2}\left(1 - 2p_1^{(0)}\right)$$

Suppose d=k: $\Pr(even|d = k) = \frac{1}{2} + \frac{1}{2}\prod_{l=0}^{k-1}\left(1 - 2p_1^{(l)}\right)$

Then d=k+1:

$$\Pr(even|d = k+1) = \Pr(even|d = k)\, p_0^{(k)} + \Pr(odd|d = k)\, p_1^{(k)}$$

$$\Pr(E|d=k+1) = \Pr(E|d=k)\left(1-p_1^{(k)}\right) + (1-\Pr(E|d=k))p_1^{(k)}$$

$$= \Pr(E|d=k)\left(1-2p_1^{(k)}\right) + p_1^{(k)}$$

$$= \left(\frac{1}{2} + \frac{1}{2}\prod_{l=0}^{k-1}\left(1-2p_1^{(l)}\right)\right)\left(1-2p_1^{(k)}\right) + p_1^{(k)}$$

$$= \frac{1}{2} + \frac{1}{2}\prod_{l=0}^{k}\left(1-2p_1^{(l)}\right)$$

proofed based on the induction. #

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}$$

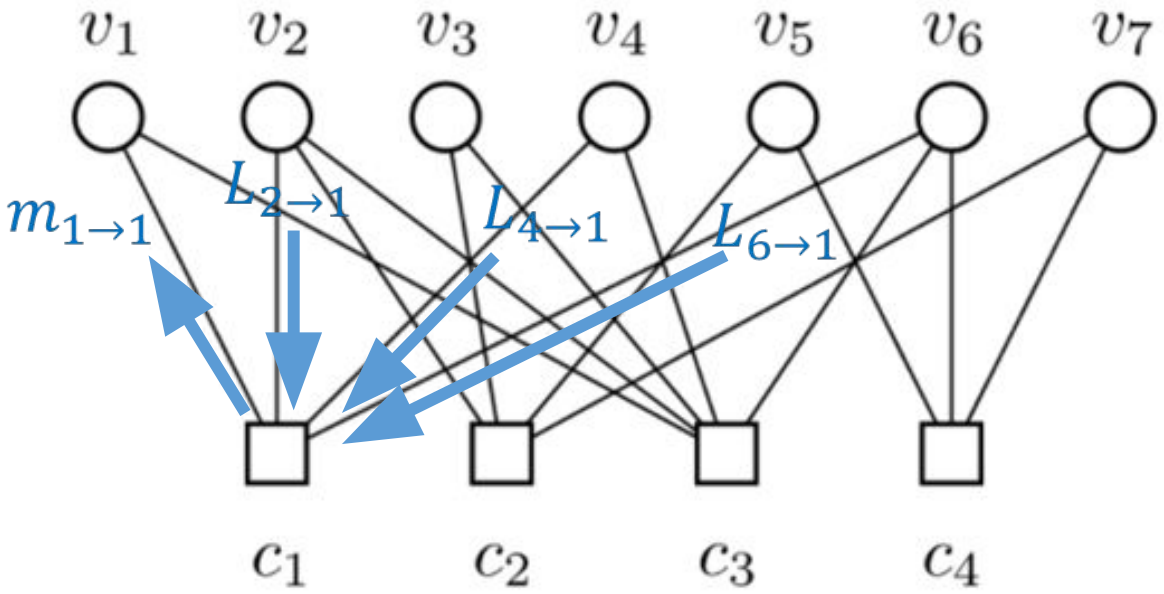
$$\tanh\left(\frac{L_{v\to c}}{2}\right) = \tanh\left(\frac{1}{2}\log\left(\frac{p_0}{p_1}\right)\right) = \frac{e^{\log\left(\frac{p_0}{p_1}\right)} - 1}{e^{\log\left(\frac{p_0}{p_1}\right)} + 1} = \frac{\frac{p_0}{p_1} - 1}{\frac{p_0}{p_1} + 1}$$

$$= \frac{p_0 - p_1}{p_0 + p_1} = p_0 - p_1 = 1 - 2p_1$$

$$\Pr\left(E | v_b \in \mathcal{N}(c_i)\backslash v_j\right) = \frac{1}{2} + \frac{1}{2}\prod_{v_b \in \mathcal{N}(c_i)\backslash v_j} \tanh\left(\frac{L_{b\to i}}{2}\right)$$

$$\tanh\left(\frac{L_{j\to i}}{2}\right) = 1 - 2\left(1 - \Pr\left(E | v_b \in \mathcal{N}(c_i)\backslash v_j\right)\right)$$

$$= -1 + 2\Pr\left(E | v_b \in \mathcal{N}(c_i)\backslash v_j\right) = \prod_{v_b \in \mathcal{N}(c_i)\backslash v_j} \tanh\left(\frac{L_{b\to i}}{2}\right)$$



The value of $m_{1\to1}$ is related to the probability that $L_{2\to1}$, $L_{4\to1}$ and $L_{6\to1}$ contain even 1s => $m_{i\to j} = L_{j\to i}$ if $L_{j\to i}$ satisfies c_i.

$$\tanh\left(\frac{m_{1\to1}}{2}\right) = \tanh\left(\frac{L_{2\to1}}{2}\right)\tanh\left(\frac{L_{4\to1}}{2}\right)\tanh\left(\frac{L_{6\to1}}{2}\right)$$

Variable node operation:

$$L_{j\to i} = L_{v_j} + \sum_{c_a \in \mathcal{N}(v_j)\backslash c_i} m_{a\to j}$$

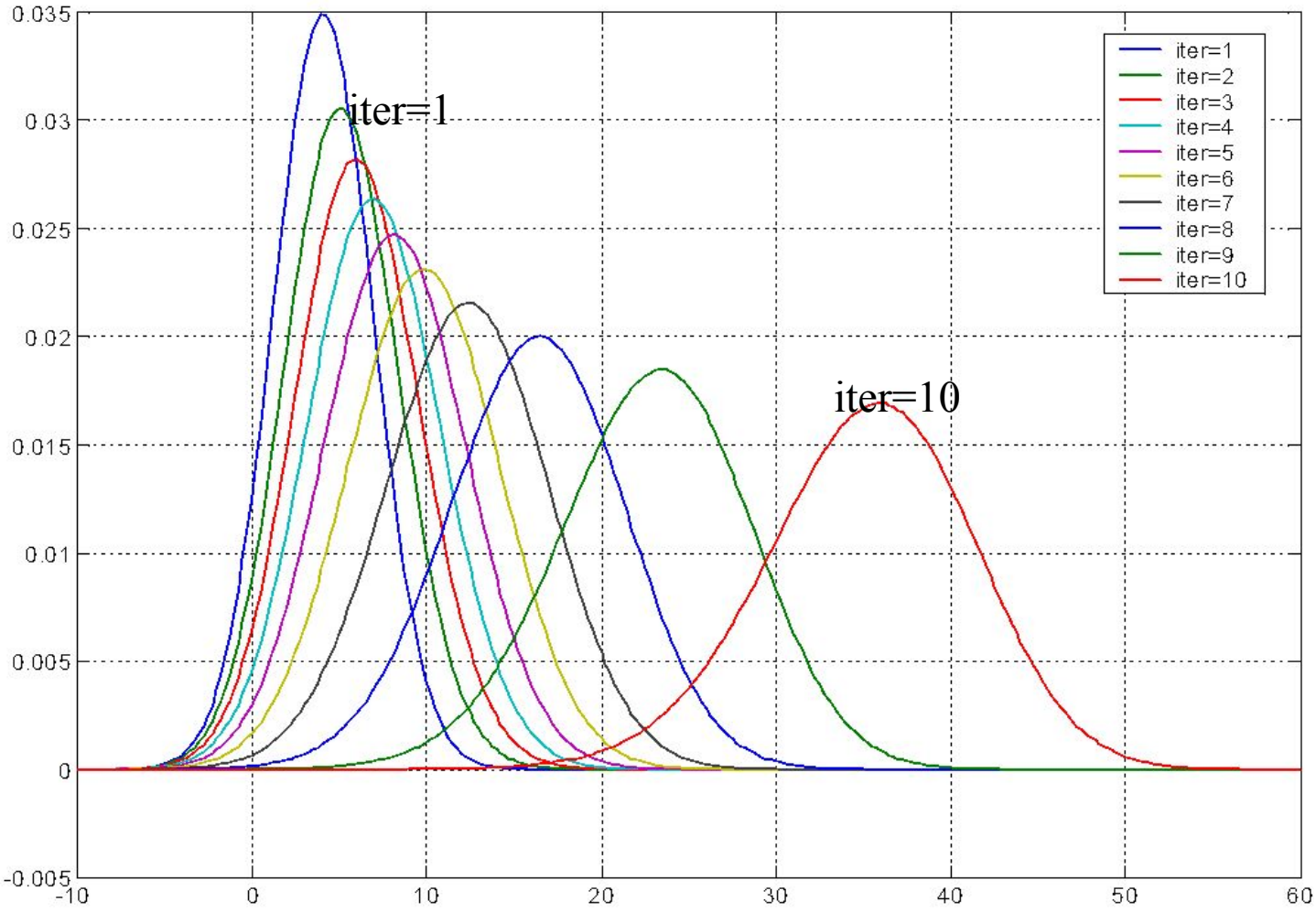
Check node operation:

$$m_{i\to j} = 2\tanh^{-1}\left(\prod_{v_b \in \mathcal{N}(c_i)\backslash v_j} \tanh\left(\frac{L_{b\to i}}{2}\right)\right)$$

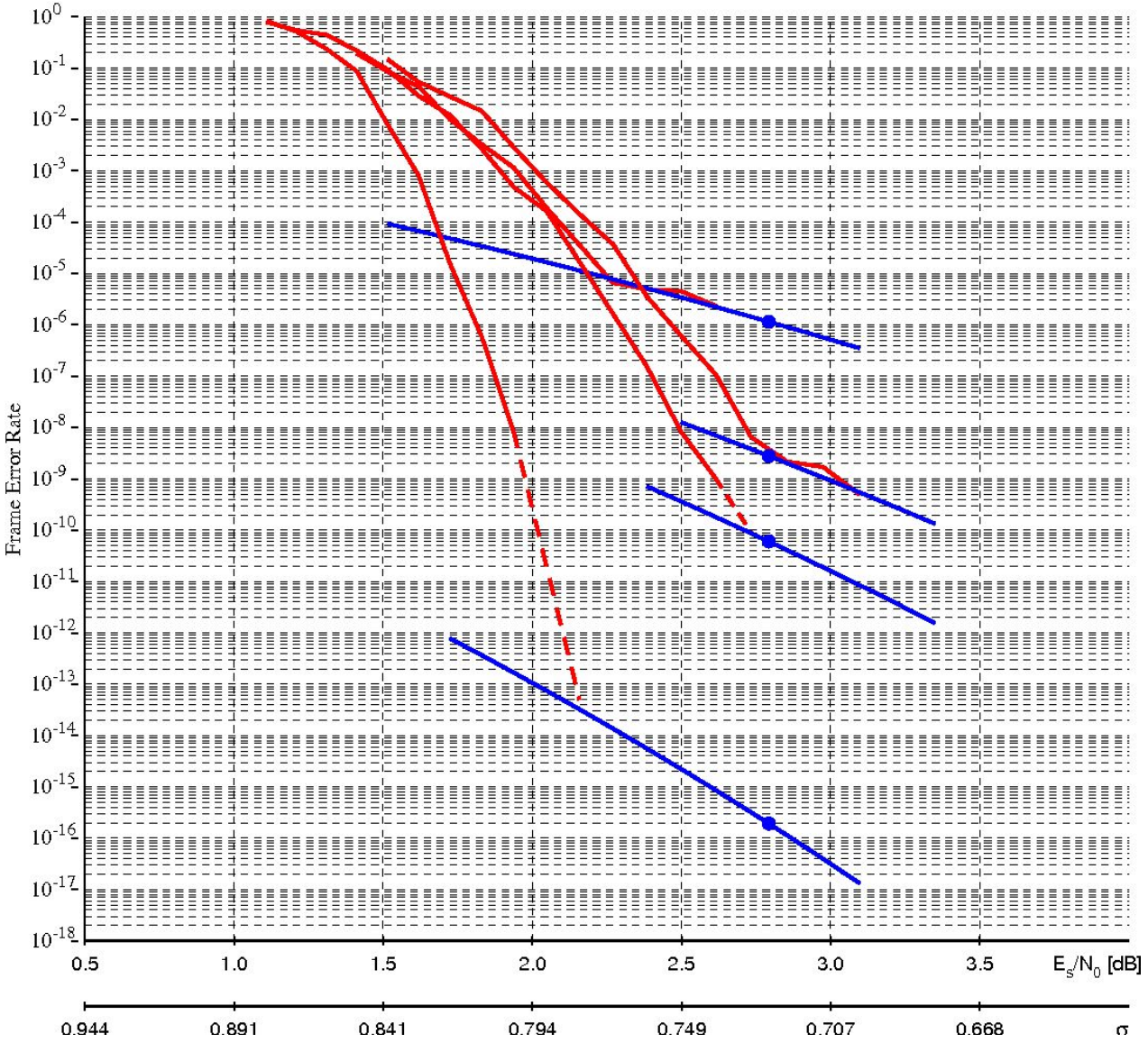
The LLR value of the APP for the j-th variable node:

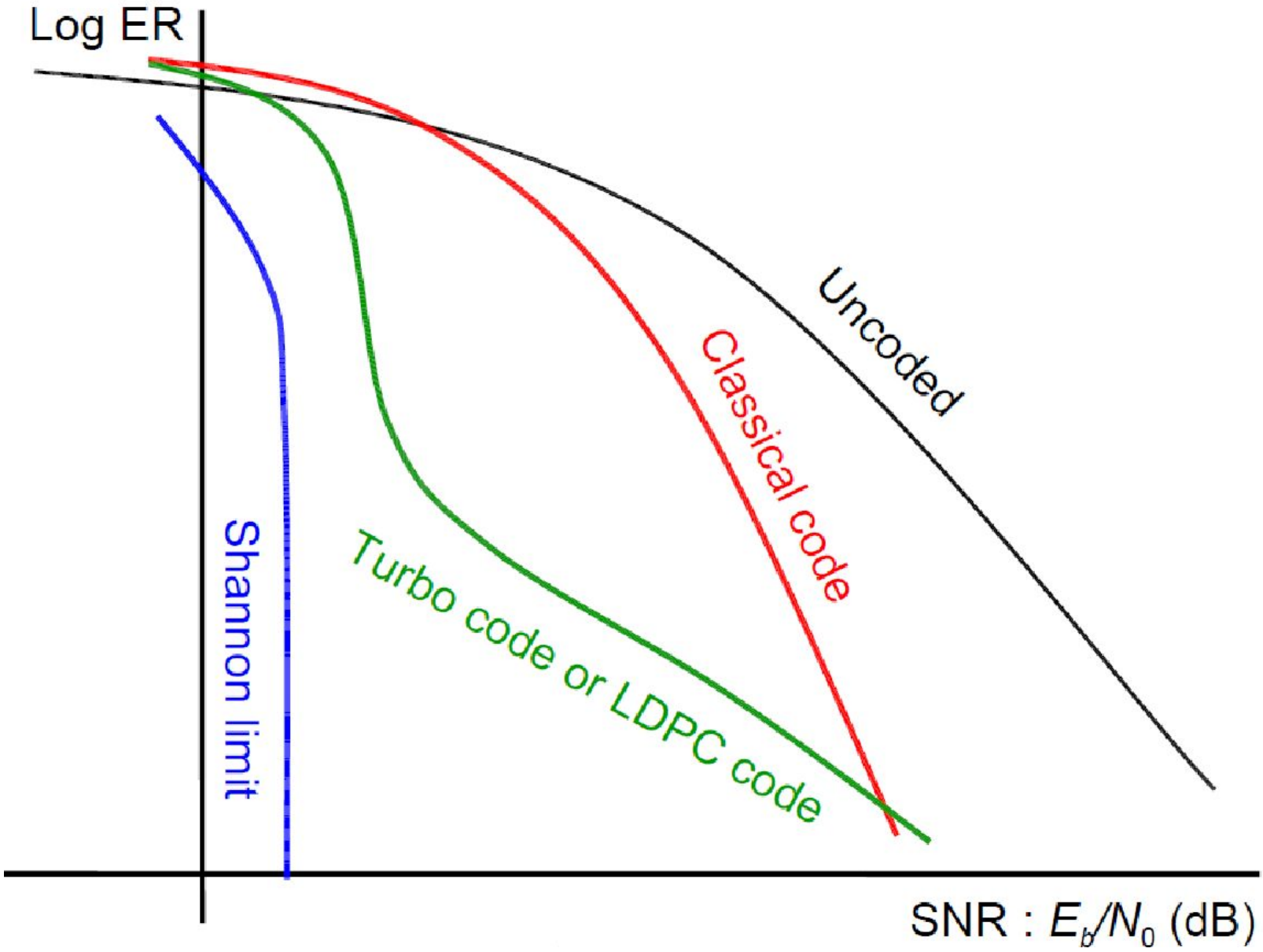
$$L_j = L_{v_j} + \sum_{i=0}^{d} m_{i\to j}$$

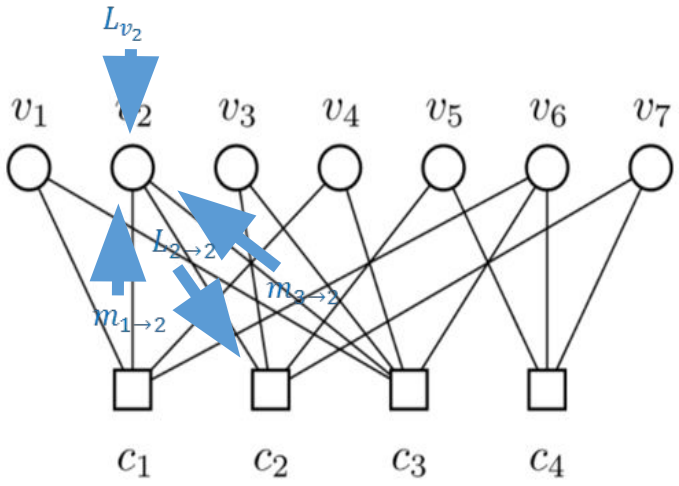
The distribution of variable-node values v.s. number of iteration:

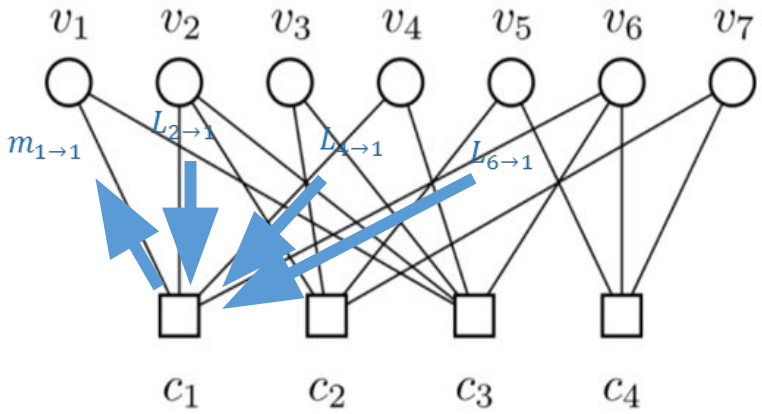


Trapping Set and Error Floor in LDPC codes




Log ER
Uncoded
Classical code
Turbo code or LDPC code
Shannon limit
SNR : E_b/N_0 (dB)





There is an assumption when SPA is introduced to replace MAP:

The sample values of individual bits are independent.

But the assumption no longer stands after the path of the passing decoding messages form a loop (cycle).

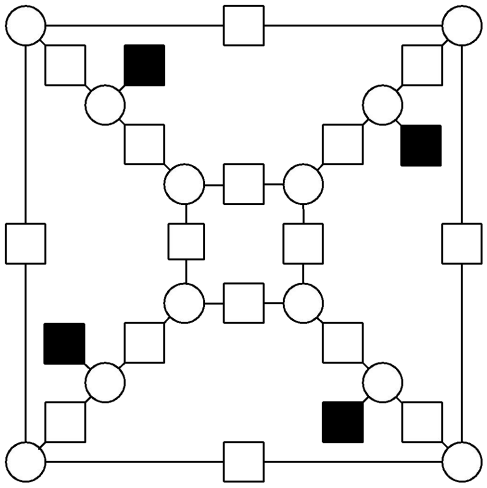
The result of SPA cannot be equal to that of MAP if the correlation between variable nodes are accumulated.

Short cycles in the Tanner graphs easily introduce poor error rate performance (high error floor).

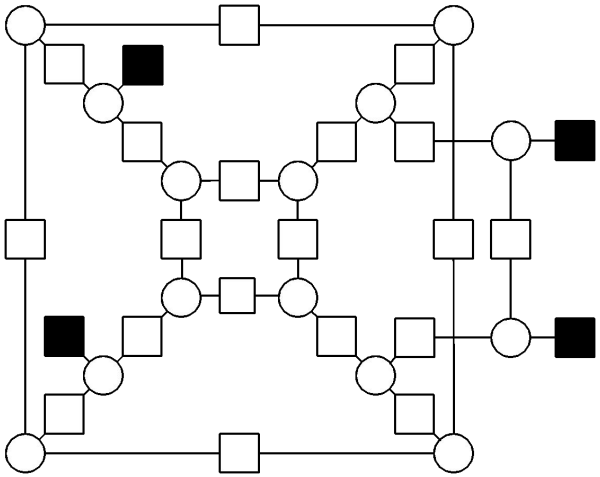
The error floor of LDPC codes is majorly due to dominate trapping sets in high SNR region.

(*a*,*b*) trapping set: *a* variable nodes are simultaneously in error and *b* check nodes connecting to these variable nodes are not satisfied.

Example: (12,4) and (14,4) trapping sets of (2640,1320) Margulis code.



(12,4)



(14,4)

The error floor lowering techniques:

Code Construction:

Construct codes with large girth (minimum length of cycle).

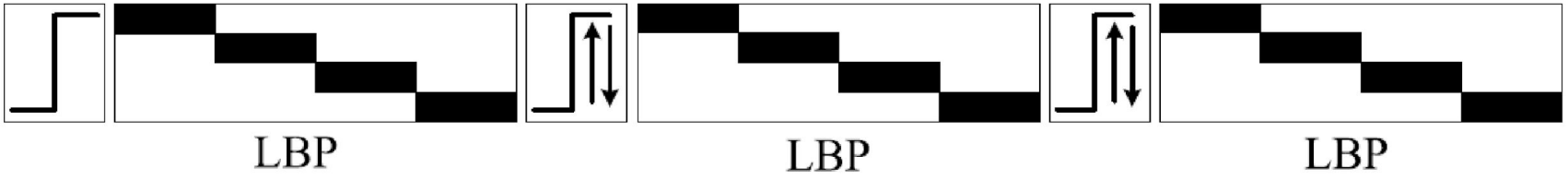
PEG: Progressive Edge Growth Algorithm.

Algebra-Assisted Construction: Reed–Solomon-based LDPC codes.

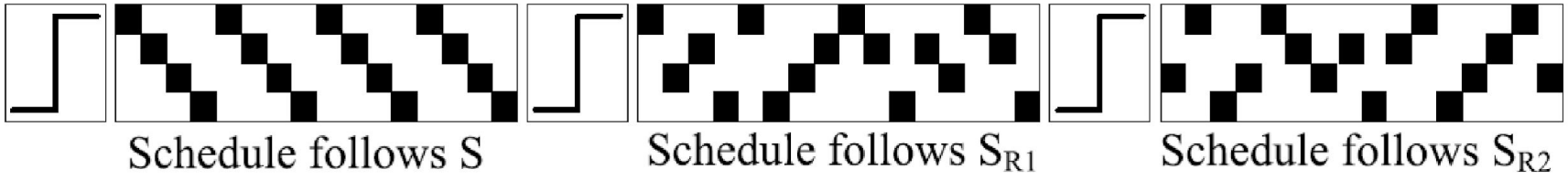
Post-Processing after regular decoding:

Erase the suspicious error bits

Backtracking + Flipping



Scheduling diversity



LDPC codes usual to specify by the VN and CN **degree-distribution polynomials**, denoted by $\lambda(X)$ and $\rho(X)$, respectively.

In a Tanner graph, the degree of a node is defined as the number of edges that are incident with the node.

$$\lambda(X) = \sum_{d=1}^{d_v} \lambda_d X^{d-1}, \rho(X) = \sum_{d=1}^{d_c} \rho_d X^{d-1}$$

λ_d : the fraction of all edges connected to degree-d VNs.

d_v : the maximum VN degree.

ρ_d : the fraction of all edges connected to degree-d CNs.

d_c : the maximum VN degree.

For (2,4)-regular LDPC code, $\lambda(X) = X, \rho(X) = X^3$.

Protograph LDPC codes, QC-LDPC codes

$$\mathbf{B}=\begin{bmatrix}1 & 1 & 1 & 0\\ 1 & 1 & 1 & 1\\ 0 & 1 & 0 & 1\end{bmatrix},\mathbf{H}=\begin{bmatrix}\mathbf{h}_{1,1} & \mathbf{h}_{1,2} & \mathbf{h}_{1,3} & \mathbf{0}_{z\times z}\\ \mathbf{h}_{2,1} & \mathbf{h}_{2,2} & \mathbf{h}_{2,3} & \mathbf{h}_{2,4}\\ \mathbf{0}_{z\times z} & \mathbf{h}_{3,2} & \mathbf{0}_{z\times z} & \mathbf{h}_{3,4}\end{bmatrix}$$

$$\mathbf{h}_{1,1}=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix},$$

$$\mathbf{h}_{1,2}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 1 & 0 & 0\end{bmatrix},\cdots,$$

$$\mathbf{h}_{3,4}=\begin{bmatrix}0 & 0 & 1\\ 1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$$