Quantum Error Correction with Girth-16 Non-Binary LDPC Codes via Affine Permutation Construction

Kenta Kasai

Institute of Science Tokyo

ISTC 2025, UCLA, August, 2025

Background (Hashing Bound Approaching Codes and Their Challenges)

- Recent studies[1] have reported that binary CSS codes constructed from $\mathbb{F}_q$-valued $(J = 2, L)$-regular LDPC codes[2] can achieve **near-hashing-bound** decoding performance over the depolarizing channel using **BP** decoding. $(q = 256)$
- The girth was upper-bounded by $2L$[1], and the coding rate was given by $R = 1 - 2J/L$.
- For codes with a low coding rate R, increasing the block length has led to a **significant error floor**.
- This was caused by BP getting trapped in **short cycles** and by **small minimum distance**: $d_{\min} < 10$.

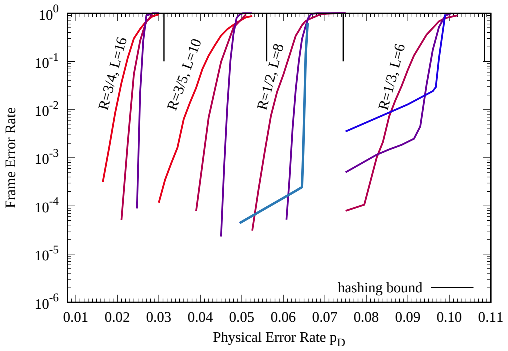


Figure: BP performance of QEC using non-binary LDPC codes over $\mathbb{F}_q$ $(q = 2^8)$.

---

[1] Komoto and Kasai, **to appear**, *npj Quantum Information*, 2025.

[2] Kasai, Hagiwara, Imai, and Sakaniwa, IEEE Trans. Information Theory, 2011.

Background (Solution for rate-$1/3$ codes)

- For codes with $R = 1/3$, even at large block lengths, a code modification[3] that **increases the minimum distance** $d_{\min} \leq 14$ and $O_n(1)$ **post-processing**[3] after BP decoding have achieved FERs as low as 10^{-5}.
- In this study, for codes with $R = 1/2$, i.e., $L = 8$, we aim to **reduce the error floor without post-processing** at large block lengths.

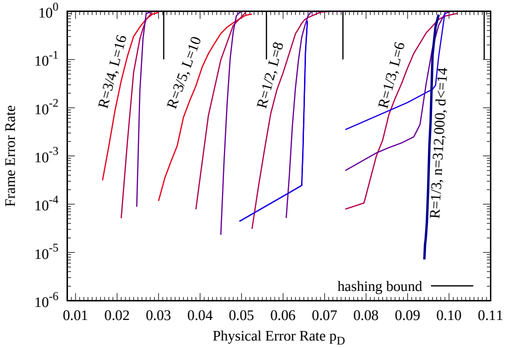
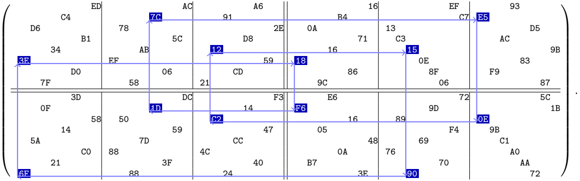


Figure: BP peformance of QEC with non-binary LDPC codes with $\mathbb{F}_q$ $(q = 2^8)$.

[3] K. Kasai, *arXiv:2506.15636*, 2025.

Method for Maximizing Girth and Distance in $J = 2$ LDPC Codes

- In LDPC codes with a parity-check matrix of column weight $J = 2$, rank-deficient cycles generate non-zero codewords.
- In general, all codewords in LDPC codes with $J = 2$ consist of unions of such cycle codewords.
- Our approach is to maximize the girth up to its upper bound of $2L = 16$ and, moreover, to increase the minimum distance.
- In QC-LDPC–based constructions[2], the girth is upper-bounded by 12; hence an alternative approach is required.



$$\begin{pmatrix} & \mathtt{7C} & & & & \mathtt{E5} \\ & & \mathtt{12} & & \mathtt{15} & \\ \mathtt{3E} & & & \mathtt{18} & & \\ & \mathtt{1D} & & \mathtt{F6} & & \\ & & \mathtt{C2} & & & \mathtt{0E} \\ \mathtt{6F} & & & & \mathtt{90} & \end{pmatrix} \begin{pmatrix} \mathtt{C8} \\ \mathtt{C8} \\ \mathtt{AA} \\ \mathtt{EE} \\ \mathtt{A7} \\ \mathtt{5F} \end{pmatrix} = \underline{0}$$

Figure: If a cycle of length 2ℓ is rank-deficient, then a cycle codeword of weight ℓ exists.

[2] Kasai, Hagiwara, Imai, and Sakaniwa, IEEE Trans. Information Theory, 2011.

Permutation-Matrix-Based Code Construction

We base our code design on the following procedure[1].

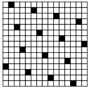
1. Prepare $P \times P$ binary **commuting** permutation matrices $\{f_i\}_{i=0}^{L/2-1}$ and $\{g_i\}_{i=0}^{L/2-1}$:
$$f_i g_j = g_j f_i \quad \text{for all } i, j.$$
2. Construct binary base matrices $\hat{H}_X$ and $\hat{H}_Z$ by arranging f_i and g_i in a cyclic manner, as illustrated below. Example with $J = 2$ and $L = 8$:
$$\hat{H}_X = \left(\begin{array}{cccc|cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$
$$\hat{H}_Z = \left(\begin{array}{cccc|cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$
With this choice, they become orthogonal: $\hat{H}_X \hat{H}_Z^{\mathsf{T}} = 0$. **The task of this study is to select $\{f_i\}$ and $\{g_i\}$ so that $\hat{H}_X$ and $\hat{H}_Z$ contain no cycles of length less than 16.**
3. Non-binarize step: choose the $\mathbb{F}_q$-valued entries of H_X and H_Z so that they have the same support (positions of nonzero entries) as $\hat{H}_X$ and $\hat{H}_Z$, respectively, and satisfy the orthogonality condition
$$H_X H_Z^{\mathsf{T}} = 0.$$

[1] Komoto and Kasai, **to appear**, *npj Quantum Information*, 2025.

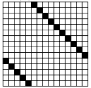
Restrictions on $\{f_i\}$ and $\{g_i\}$ in Code Construction

1. **Commutativity** condition for orthogonality:
   $$f_i g_j = g_j f_i$$
2. Non-commutativity condition:
   Any 2×3 commuting submatrix always contains a 12-cycle. Therefore, $\{f_i\}$ and $\{g_i\}$ must be **non-commuting within each set**.
3. Use affine permutation matrices (**APM**): CPM vaiolates non-commutativity condition.
   Attempt to construct codes with girth 16 with general permutation matrices failed due to huge search space and high complexity.

APM
$c(r) = 4r + 1$



CPM
$c(r) = r + 5$



$c(r), r \in \mathbb{Z}_P, P = 15$

$$\hat{H}_X = \left(\begin{array}{cccc|cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array} \right)$$

$$\hat{H}_Z = \left(\begin{array}{cccc|cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array} \right)$$

	f_0	f_1	f_2	f_3	g_0	g_1	g_2	g_3
f_0	–	0	0	0	1	1	1	1
f_1	0	–	0	0	1	1	1	1
f_2	0	0	–	0	1	1	1	1
f_3	0	0	0	–	1	1	1	1
g_0	1	1	1	1	–	0	0	0
g_1	1	1	1	1	0	–	0	0
g_2	1	1	1	1	0	0	–	0
g_3	1	1	1	1	0	0	0	–

Legend: 1 commute, 0 non-commute, – diagonal.

[4] M. P. C. Fossorier, IEEE Transactions on Information Theory, 2004.

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc|cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc|cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$

$f_0(x) = 3151X + 7075,\quad f_1(x) = 9451X + 6495,\quad f_2(x) = 7351X + 1295,\quad f_3(x) = 10501X + 3540$
$g_0(x) = 6301X + 5178,\quad g_1(x) = 5041X + 9360,\quad g_2(x) = X + 4584,\quad g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600,\ J = 2,\ L = 8,\ R = 1/2,\ n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc||cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array} \right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc||cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array} \right).$$

$f_0(x) = 3151X + 7075, \quad f_1(x) = 9451X + 6495, \quad f_2(x) = 7351X + 1295, \quad f_3(x) = 10501X + 3540$

$g_0(x) = 6301X + 5178, \quad g_1(x) = 5041X + 9360, \quad g_2(x) = X + 4584, \quad g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc||cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc||cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$

$f_0(x) = 3151X + 7075$, $f_1(x) = 9451X + 6495$, $f_2(x) = 7351X + 1295$, $f_3(x) = 10501X + 3540$
$g_0(x) = 6301X + 5178$, $g_1(x) = 5041X + 9360$, $g_2(x) = X + 4584$, $g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc||cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc||cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$

$f_0(x) = 3151X + 7075$, $f_1(x) = 9451X + 6495$, $f_2(x) = 7351X + 1295$, $f_3(x) = 10501X + 3540$
$g_0(x) = 6301X + 5178$, $g_1(x) = 5041X + 9360$, $g_2(x) = X + 4584$, $g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc|cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array} \right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc|cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array} \right).$$

$f_0(x) = 3151X + 7075$, $f_1(x) = 9451X + 6495$, $f_2(x) = 7351X + 1295$, $f_3(x) = 10501X + 3540$
$g_0(x) = 6301X + 5178$, $g_1(x) = 5041X + 9360$, $g_2(x) = X + 4584$, $g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc||cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc||cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$

$f_0(x) = 3151X + 7075,\quad f_1(x) = 9451X + 6495,\quad f_2(x) = 7351X + 1295,\quad f_3(x) = 10501X + 3540$

$g_0(x) = 6301X + 5178,\quad g_1(x) = 5041X + 9360,\quad g_2(x) = X + 4584,\quad g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc||cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc||cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$

$f_0(x) = 3151X + 7075, \quad f_1(x) = 9451X + 6495, \quad f_2(x) = 7351X + 1295, \quad f_3(x) = 10501X + 3540$
$g_0(x) = 6301X + 5178, \quad g_1(x) = 5041X + 9360, \quad g_2(x) = X + 4584, \quad g_3(x) = 7561X + 5784$

Progressive Random Code Construction Algorithm (target girth 16)

Select APMs $\{f_i\}_{i=0}^{L/2-1}, \{g_i\}_{i=0}^{L/2-1}$ one by one by generating random candidates.
For each candidate, check the following three conditions:

- **Commutativity** between f_i and g_j,
- **Non-commutativity** within $\{f_i\}$ and within $\{g_i\}$,
- **No cycles shorter than 16** in base matrices $\hat{H}_X$ and $\hat{H}_Z$.

If a candidate fails any of these conditions, reject it and generate a new one.

Example: APMs for $P = 12,600$, $J = 2$, $L = 8$, $R = 1/2$, $n = 806,400$, girth$= 16$

$$\hat{H}_X = \left(\begin{array}{cccc||cccc} f_0 & f_1 & f_2 & f_3 & g_0 & g_1 & g_2 & g_3 \\ f_3 & f_0 & f_1 & f_2 & g_3 & g_0 & g_1 & g_2 \end{array}\right),$$

$$\hat{H}_Z = \left(\begin{array}{cccc||cccc} f_0^{-1} & f_3^{-1} & f_2^{-1} & f_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} & g_1^{-1} \\ f_1^{-1} & f_0^{-1} & f_3^{-1} & f_2^{-1} & g_1^{-1} & g_0^{-1} & g_3^{-1} & g_2^{-1} \end{array}\right).$$

$f_0(x) = 3151X + 7075$, $f_1(x) = 9451X + 6495$, $f_2(x) = 7351X + 1295$, $f_3(x) = 10501X + 3540$
$g_0(x) = 6301X + 5178$, $g_1(x) = 5041X + 9360$, $g_2(x) = X + 4584$, $g_3(x) = 7561X + 5784$

BP decoding

- Non-binarized matrices: H_X and H_Z
- Measure syndromes:

$$\boldsymbol{s} = H_Z \boldsymbol{x}, \quad \boldsymbol{t} = H_X \boldsymbol{z}$$

- BP iteratively and **simultaneously estimates** estimates $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{z}}$ at each iteration.
- Error correction is regarded as successful if and only if

$$\boldsymbol{x} + \hat{\boldsymbol{x}} \in C_X^{\perp} \text{ and } \boldsymbol{z} + \hat{\boldsymbol{z}} \in C_Z^{\perp}.$$

- In our experiments, whenever decoding was successful, we always observed $\boldsymbol{x} + \hat{\boldsymbol{x}} = 0$ and $\boldsymbol{z} + \hat{\boldsymbol{z}} = 0$. On the other hand, when decoding failed, $w_H(\boldsymbol{x} + \hat{\boldsymbol{x}})$ and $w_H(\boldsymbol{z} + \hat{\boldsymbol{z}})$ are propotional to n. This indicates that the observed errors are not due to the error floor, and therefore **no additional post-processing after BP is required**.

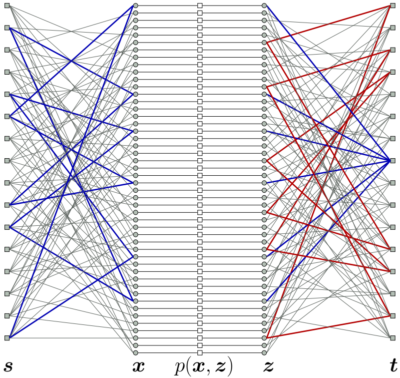


Figure: Factor graph of BP.

Minimum Distance Estimation: Upper Bound

- Minimum distance $d_{\min}$

$$
\begin{aligned}
d_X &= \min\left\{w_H(\mathbf{c}) \mid \mathbf{c} \in C_X \backslash C_Z^{\perp}\right\} \\
d_Z &= \min\left\{w_H(\mathbf{c}) \mid \mathbf{c} \in C_Z \backslash C_X^{\perp}\right\} \\
d_{\min} &= \min\left\{d_X, d_Z\right\},
\end{aligned}
$$

- We calculate the bit-level weight distribution of codewords formed from the shortest cycles.

1. Enumerate all shortest cycle codewords in $C_X \setminus C_Z^{\perp}$ and $C_Z \setminus C_X^{\perp}$.
2. $A_X(w)$ and $A_Z(w)$: Count the number of such codewords of weight w, respectively.
3. From the table, we observe that the minimum bitwise weight among these codewords is 14.

Prop.			Conv.		
w	$A_X(w)$	$A_Z(w)$	w	$A_X(w)$	$A_Z(w)$
14	0	2	9	0	7
15	7	4	10	3	10
16	19	16	11	23	36
17	64	74	12	115	117
18	180	243	13	326	393
19	517	545	14	1012	976
20	1395	1394	15	2512	2413
21	3198	3361	16	5570	5426
22	6557	6580	17	10975	10905
23	12432	12604	18	19321	18577
24	22065	22441	19	31644	30583
25	36562	37114	20	46466	44906
26	56105	57037	21	62938	60967
27	80796	81754	22	78365	75967
28	108252	109732	23	89540	86380
29	136363	137293	24	92775	90397
30	161125	162252	25	89613	87421
31	177894	179259	26	79558	77923
32	184555	185588	27	65163	63004
33	179963	180763	28	48834	47345
34	164748	166054	29	33761	33019
35	141673	142321	30	21240	20708
36	113975	115282	31	12677	11926
37	86629	87555	32	6664	6374
38	61398	62078	33	3160	3139
39	41178	41603	34	1436	1340
40	25493	25983	35	540	527
41	14795	15334	36	205	194
42	8001	8279	37	67	61
43	4183	4287	38	17	19
44	1946	1997	39	5	3
45	858	887	40	0	2
46	338	340	41	0	0
47	127	150	42	0	0
48	46	46	43	0	0
49	8	19	44	0	0
50	4	7	45	0	0
51	1	1	46	0	0
52	0	1	47	0	0
50	4	7	38	17	19
51	1	1	39	5	3
52	0	1	40	0	2
total	255× 7192	255× 7256	total	255× 3155	255× 3063

Numerical Result:

- For the code with $R = 1/2$ and $L = 8$, we reduced the error floor using only BP decoding without any post-processing.
- No error floor has been observed down to a frame error rate of at least 10^{-6}.
- The block length is close to one million, which is extremely large.

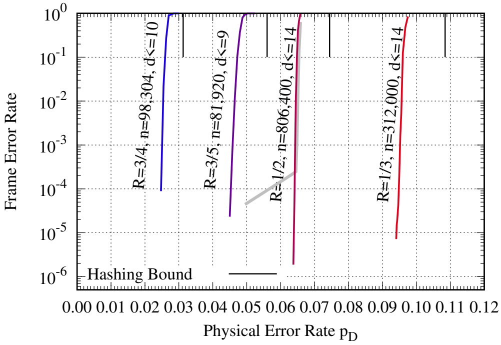


Figure: Comparison: Conventional method (grey) with proposed method (red) for $R = 1/2$.

Conclusions and Future work

- We successfully constructed long quantum LDPC codes with girth 16 and rate $1/2$, whose minimum distance does not exceed 14, achieving a deep error floor without post-processing.
- We plan to further increase the minimum distance by modifying the non-binary entries of cycle codewords associated with 16-cycles.
- We plan to analyze the error floor using weight distribution.
- We also aim to further lower the error floor by applying post-processing, although observing the error floor remains challenging.

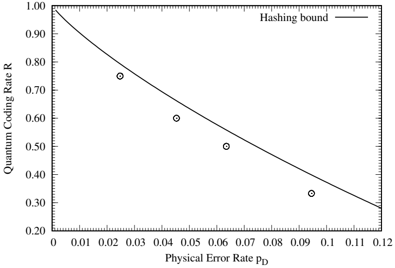


Figure: Physical Error Rate required for FER$=10^{-4}$ vs. Coding Rate