Optimization tools for distance-preserving flag fault-tolerant error correction

Balint Pato,[1, 2, *] Theerapat Tansuwannont,[1, 2, †] Shilin Huang,[1, 2, ‡] and Kenneth R. Brown[1, 2, 3, 4, §]

[1]*Duke Quantum Center, Duke University, Durham, NC 27701, USA*
[2]*Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA*
[3]*Department of Physics, Duke University, Durham, NC 27708, USA*
[4]*Department of Chemistry, Duke University, Durham, NC 27708, USA*

Lookup table decoding is fast and distance-preserving, making it attractive for near-term quantum computer architectures with small-distance quantum error correcting codes. In this work, we develop several optimization tools which can potentially reduce the space and time overhead required for flag fault-tolerant error correction (FTEC) with lookup table decoding on Calderbank-Shor-Steane (CSS) codes. Our techniques include the compact lookup table construction, the Meet-in-the-Middle technique, the adaptive time decoding for flag FTEC, the classical processing technique for flag information, and the separated X and Z counting technique. We evaluate the performance of our tools using numerical simulation of hexagonal color codes of distances 3, 5, 7, and 9 under circuit-level noise. Combining all tools can result in more than an order of magnitude increase in pseudothreshold for the hexagonal color code of distance 9, from $(1.34 \pm 0.01) \times 10^{-4}$ to $(1.42 \pm 0.12) \times 10^{-3}$.

I. INTRODUCTION

Inside a future large-scale quantum computer, there will be a continuous battle against unwanted interactions with the environment. The main goal of fault-tolerant quantum error correction (FTEC) protocols [1] is to create a robust channel to transfer quantum information from the past to the future. The threshold theorem states that it is possible to suppress the logical error rate to an arbitrarily small value given that the physical error rate is below the accuracy threshold [2–5]. It is essential to reduce both space and time overhead (the numbers of qubits and gates) for scalable quantum computing as decreasing logical error rates requires increasing overhead [6–9], and the current, leading proposals for FTEC schemes have daunting requirements [10].

An FTEC scheme is designed to be robust against propagating errors that emerge from faulty gates during the execution of the protocol even if the physical noise is assumed to be uncorrelated or weakly correlated. The scheme also has to protect against ancilla preparation and measurement errors, usually through repeated syndrome measurements. For an $[\![n, k, d]\!]$ stabilizer code [11] which encodes k logical qubits into n physical qubits and has distance d, Shor's solution [1] was to utilize cat-state ancilla register that requires w ancilla qubits and $(d + 1)^2/4$ rounds of syndrome measurements, where w is the maximum weight of the stabilizer generators. In Steane-style syndrome extraction [12], the ancilla register requires n qubits and is encoded with the same quantum error correcting code (QECC) as the data qubits. Similarly, in Knill-style error correction [13], the ancilla

register consists of two blocks of n qubits encoded in the same QECC as the data qubits.

In contrast to complex ancilla structure, for some specific families of codes, stabilizer generators can be fault-tolerantly measured using bare ancillas [14–16]. For a general stabilizer code, however, generator measurements with bare ancillas might not be possible. A series of works aiming to reduce the size of the ancilla register resulted in increasingly lighter-weight constructions [17, 18], which also led to the flag FTEC schemes for perfect codes of distance 3 that use only 2 ancillas per generator [19]. The flag schemes later generalized to arbitrary codes of distance d require $d + 1$ ancillas per generator [20], while the schemes for some specific families of codes require fewer [21–26].

FTEC schemes based on extraction circuits with cat states and flag FTEC schemes both require repetition of syndrome measurements that can result in a large number of gates. Adaptive syndrome measurement schemes in which the subsequent measurement procedures depend on the previous syndrome measurement outcomes have been explored to reduce the number of rounds required for FTEC schemes with Shor-type extraction circuits [27–29].

In a non-ideal FTEC protocol, faults can occur at any gate on any round of the syndrome measurements. The only information about the error on data qubits that we can obtain is a sequence of error syndromes, and we want to find an appropriate recovery operator from this information. An ideal strategy would be using all syndrome bits from all rounds, that is, the whole measurement outcomes in space-time. For some codes with a nice structure such as surface codes, an efficient space-time decoder exists [30]. However, constructing a space-time decoder for a general stabilizer code is not simple. To simplify the problem, we will consider an error decoder which is composed of two parts: the space and the time decoders. Under the assumption that the syndrome measurements can be faulty, the time decoder finds a round of syndrome measurements that has no faults and give a cor-

rect syndrome. The space decoder then uses the correct syndrome to find a recovery operator.

Conventionally, flag FTEC uses a lookup table decoder as a space decoder, and relies on Shor-style repeated syndrome measurements as a time decoder. These decoders have pros and cons. The lookup table decoder is fast and distance-preserving. However, building a lookup table requires an exhaustive search on all possible fault combinations up to a certain number of faults, and the table requires a lot of memory to store. Thus, it might not work well with a code of high distance (unless code concatenation is applied). The Shor-style time decoder is simple and compatible with any space decoder. However, the large time overhead required in the repetition can result in a lower threshold.

In this work, we build several optimization tools for both space and time decoders for flag FTEC so that it becomes more practical. Most of our tools are applicable to a general stabilizer code, but we primarily focus on a self-dual CSS code in which the number of physical qubits is odd, the number of logical qubits is 1, and logical X and Z operators are transversal for simplicity. Our main results are the following: (1) We develop a technique to build a lookup table more efficiently. Our compact lookup table for a self-dual CSS code requires 75% less memory footprint compared to a lookup table for a generic stabilizer code. Our method also efficiently verifies whether a given configuration of the flag circuits preserves the code distance. The development also leads to a notion of fault code which can be useful in error sampling for the circuit-level noise model. (2) We introduce the Meet-in-the-Middle (MIM) technique which can help the lookup table decoder to correct faults more than the number of errors correctable by the underlying code. Although the correction is not always successful, the higher success probability can significantly increase the pseudothreshold in our simulation. (3) We generalize the adaptive syndrome measurement scheme from [29] to flag FTEC and introduce one-tailed and two-tailed adaptive time decoders, which are useful in different circumstances. We also develop a classical processing technique on flag information that makes our FTEC protocols compatible with any fault-tolerant Clifford computation. (4) We use our optimization tools and perform numerical simulations on the hexagonal color codes [31] of distances 3, 5, 7, and 9. The results show that each of our tools can significantly reduce the logical error rates and increase the pseudothreshold for each code while preserving the code distance. For the hexagonal color code of distance 9, the pseudothreshold is improved by one order of magnitude, from $(1.34 \pm 0.01) \times 10^{-4}$ to $(1.42 \pm 0.12) \times 10^{-3}$, when all techniques are applied.

This paper is organized as follows. In Section II, we define the noise model in this work, review flag FTEC, and provide definitions of fault-tolerant error correction. In Section III, we develop optimization tools for space decoder, including an efficient method to build a compact lookup table and the MIM technique. In Section IV,

we develop optimization tools for time decoder, including the one-tailed and two-tailed adaptive time decoder, and other extended techniques for CSS codes. In Section V, we provide numerical results for the hexagonal color codes and observe the effects of the MIM, the adaptive time decoding, and the separated X and Z counting techniques on the logical error rates. We discuss and conclude our results in Section VI.

II. BACKGROUND

Quantum systems are fragile and can lose their properties easily when interacting with the environment. To protect quantum information, one can use a quantum error correcting code (QECC) to encode the quantum data. Quantum error correction (QEC) is a process that identifies an error when it occurs, then applies an appropriate error correction (EC) operator to remove the error. However, quantum operations in the process can be faulty and may introduce more errors to the system. For this reason, we want to make sure that the QEC process is *fault tolerant*; that is, an imperfect QEC process can correct errors faster than causing new ones.

In this section, we first describe the noise model that will be used in this work and provide the conventional definition of fault-tolerant error correction in Section II A. We then review flag FTEC and provide a revised definition of fault-tolerant error correction which is more suitable for flag FTEC in Section II B.

A. Noise model and conventional definition of fault-tolerant error correction

An $[[n, k, d]]$ stabilizer code [11] encodes k logical qubits using n physical qubits and can correct up to $\tau = \lfloor (d - 1)/2 \rfloor$ errors, where d is the code distance. A stabilizer code is described by a *stabilizer group*, an Abelian group generated by $r = n - k$ commuting Pauli operators whose elements are called stabilizers. The coding subspace is the simultaneous $+1$ eigenspace of all elements in the stabilizer group.

The QEC process for a stabilizer code can be done by first measuring the eigenvalues of all stabilizer generators. An r-bit string of measurement outcomes is called the error syndrome (where bits 0 and 1 refer to $+1$ and -1 eigenvalues of each generator). An example of a circuit for measuring an eigenvalue of a stabilizer generator is displayed in Fig. 1. After the syndrome (or the sequence of syndromes) is obtained, an appropriate recovery operator will be found by a mapping called *error decoder*, then the recovery operator will be applied to the data qubits. For Calderbank-Shor-Steane (CSS) codes [12, 32], it is possible to correct X- and Z-type errors separately. For a self-dual CSS code (whose X-type and the Z-type stabilizer generators are of the same form), similar error
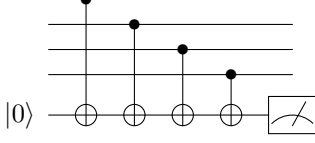
FIGURE 1: A syndrome extraction circuit with bare ancilla for measuring a stabilizer generator of the form $ZZZZ$.

decoders can be used for both X-type and Z-type error corrections.

Note that there are many Pauli operators that correspond to the same error syndrome. If the recovery operator from the decoder and the error that occurred to the codeword are *logically equivalent*, i.e., they are the same operator up to a multiplication of some stabilizer, the QEC process succeeds. However, if the recovery operator from the decoder and the error that occurred to the codeword are off by a multiplication of some nontrivial logical operator, the QEC process fails.

If all gates in the syndrome measurement process described above are perfect, a stabilizer code of distance d should be able to correct up to τ errors as desired. However, the process above is not fault-tolerant under the circuit-level depolarizing noise. This is because a single faulty gate may lead to an error that can propagate to other data qubits, causing an error of weight higher than τ that might not be correctable. For this reason, a better set of syndrome extraction circuits is required to make sure that the QEC process is fault tolerant.

Before defining fault-tolerant error correction, let us consider the noise model and error propagation. In this work, we use the *circuit-level depolarizing noise model*. Here we assume that any quantum operation can be decomposed into a series of one- and two-qubit gates, and single-qubit preparation in the $|0\rangle$ state and measurements in the Z-basis. An imperfect operation is modeled as a perfect operation followed by a Pauli operator on the support of the perfect operation, referred to as a *fault*. In this error model, every single-qubit gate is followed by a single-qubit Pauli operator $P \in \{X, Y, Z\}$ with probability $p/3$ each, and every two-qubit gate is followed by a two-qubit Pauli operator $P_1 \otimes P_2 \in \{I, X, Y, Z\}^{\otimes 2} \setminus \{I \otimes I\}$ with probability $p/15$ each. In addition, a single-qubit preparation and measurement can also be faulty; this is modeled be a bit-flip channel after a single-qubit preparation or before a single-qubit measurement with error probability p. Because any error on w qubits can be written as a linear combination of Pauli operators of weight w (where w is the number of non-identity factors of a Pauli operator), if we can correct any Pauli operator of weight w, we can also correct any type of errors on w qubits.

When an error occurs, it may be transformed into another type of error or propagated to other data qubits

depending on subsequent operations. For example, Hadamard and CNOT gates can transform errors as follows:

$$
\begin{aligned}
H: & \quad X \mapsto Z, & Z \mapsto X, \\
\mathrm{CNOT}: & \quad XI \mapsto XX, & ZI \mapsto ZI, \\
& \quad IX \mapsto IX, & IZ \mapsto ZZ.
\end{aligned}
$$

Because of this, an error on a single qubit that occurs during the syndrome measurement can propagate and become an error of higher weight at the end. Possible resulting errors depend on the properties of the syndrome extraction circuits, including the number of ancilla qubits and the gate orderings.

One way to define fault-tolerant error correction (FTEC) is using the definition proposed by Aliferis, Gottesman, and Preskill:

Definition 1. *Fault-tolerant error correction [5]*
Let $t \leq \lfloor (d-1)/2 \rfloor$ where d is the distance of a stabilizer code. An error correction protocol is t-fault tolerant if the following two conditions are satisfied:

1. *Error correction correctness property (ECCP): For any input codeword with error of weight r, if s faults occur during the protocol with $r + s \leq t$, ideally decoding the output state gives the same codeword as ideally decoding the input state.*

2. *Error correction recovery property (ECRP): If s faults occur during the protocol with $s \leq t$, regardless of the weight of the error on the input state, the output state differs from any valid codeword by an error of weight at most s.*

When a QEC protocol satisfies Definition 1, it is guaranteed that the output error will have weight $\leq t$ whenever the weight of the input error plus the total number of faults in the protocol is $\leq t$. This means that if the next round of QEC has no faults, it can always correct the output error from the current round. Normally, we would like to construct an FTEC protocol in which t is as close to $\tau = \lfloor (d-1)/2 \rfloor$. If $t = \tau$, we say that *the FTEC protocol preserves the code distance*.

B. Flag technique and revised definition of fault-tolerant error correction

Before describing the flag technique for FTEC, let us consider a well-known Shor FTEC [1] applied to a stabilizer code of distance d. In this scheme, a stabilizer generator of weight w is measured using a cat state of the form $\frac{1}{\sqrt{2}}(|0\rangle^{\otimes w} + |1\rangle^{\otimes w})$ and transversal CNOT gates; see Fig. 2. A circuit of this kind will be called a *Shor syndrome extraction circuit*. When the cat state is prepared fault-tolerantly, a single fault in the circuit can lead to an error of weight no more than 1 on the data qubits, so the
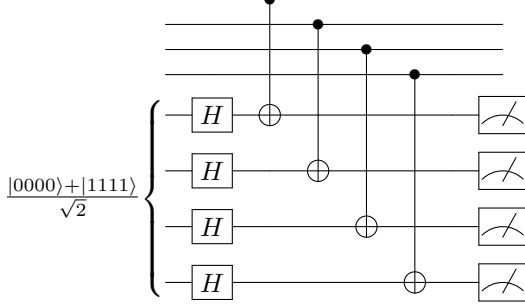
FIGURE 2: A Shor syndrome extraction circuit for measuring a stabilizer generator of the form $ZZZZ$.

FIGURE 3: A flag circuit for measuring a stabilizer generator of the form $ZZZZ$.

set of all possible errors arising from up to t faults is exactly the same as a set of all possible errors on $\leq t$ qubits in this case. Therefore, any syndrome can uniquely identify the error (up to a multiplication of some stabilizer) when the number of faults in the protocol is $\leq t$.

One drawback of the Shor syndrome extraction circuit is that the number of required ancilla qubits is equal to the maximum weight of the stabilizer generators. Also, fault-tolerant preparation of the ancilla cat state requires verification [1] or Divincenzo-Aliferis ancilla decoding circuit [17], which requires additional space and time overhead. One possible technique that can reduce the number of required ancillas for FTEC is the flag technique [19], in which each syndrome extraction circuit uses one ancilla qubit to keep the syndrome measurement outcome and a few flag ancillas to find a location that a fault might have occurred. A circuit of this kind will be called a *flag circuit*; See Fig. 3 for an example. The flag measurement outcomes give extra information that can be used to partition set of all possible errors from a certain number of faults. Therefore, it is possible to distinguish between two non-equivalent errors that correspond to the same syndrome if the flag measurement outcomes associated with each error are different, making error correction easier.

Here we define fault combination, fault set, and distinguishability of a fault set as follows:

Definition 2. *Fault combination, combined data error, and cumulative flag vector [26]*

A fault combination $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_r\}$ *is a set of r faults $\lambda_1, \lambda_2, \cdots, \lambda_r$. Suppose that the Pauli error due to the fault λ_i can propagate through the circuit and lead to data error $E(\lambda_i)$ and flag vector $\vec{f}(\lambda_i)$. The combined data error $\mathbf{E}(\Lambda)$ and cumulative flag vector $\vec{\mathbf{f}}(\Lambda)$ corresponding to the fault combination Λ are,*

$$\mathbf{E}(\Lambda) = \prod_{i=1}^{r} E(\lambda_i), \tag{1}$$

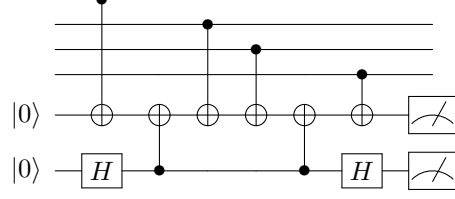$$\vec{\mathbf{f}}(\Lambda) = \sum_{i=1}^{r} \vec{f}(\lambda_i) \ (\text{mod } 2). \tag{2}$$

Definition 3. *Distinguishable fault set [26]*

Let $\mathcal{S}$ be the stabilizer group of a stabilizer code, and let the fault set $\mathcal{F}_t$ *denote the set of all possible fault combinations arising from up to t faults during the measurement of stabilizer generators of $\mathcal{S}$. We say that $\mathcal{F}_t$ is* distinguishable *if for any pair of fault combinations $\Lambda_p, \Lambda_q \in \mathcal{F}_t$, at least one of the following conditions is satisfied:*

1. $\vec{s}(\mathbf{E}(\Lambda_p)) \neq \vec{s}(\mathbf{E}(\Lambda_q))$, *or*

2. $\vec{\mathbf{f}}(\Lambda_p) \neq \vec{\mathbf{f}}(\Lambda_q)$, *or*

3. $\mathbf{E}(\Lambda_p) = \mathbf{E}(\Lambda_q) \cdot M$ *for some stabilizer $M \in \mathcal{S}$,*

where $\vec{s}(\mathbf{E})$ is the error syndrome of a combined error $\mathbf{E}$. Otherwise, we say that $\mathcal{F}_t$ is indistinguishable.

Note that the cases of faulty flag qubit measurements are included when the fault set is calculated for verifying fault set distinguishability (see Section III A). Having a distinguishable fault set is a key to successful error decoding. Given a set of syndrome extraction circuits (with or without flags), we can calculate the fault set $\mathcal{F}_t$ and check whether it is distinguishable. If it is, all possible errors arising from up to t faults that correspond to the same syndrome and cumulative flag vector are always logically equivalent. Therefore, if the syndrome measurements give a syndrome $\vec{s}$ and a cumulative flag vector $\vec{\mathbf{f}}$, we can pick any error that corresponds to the pair $(\vec{s}, \vec{\mathbf{f}})$ to be a recovery operator. Using this idea, a decoding table and an FTEC protocol can be constructed.

With the notion of fault distinguishablity, it is possible to further generalize the definition of FTEC as follows:

Definition 4. *Fault-tolerant error correction (revised) [26]*

Let $t \leq \lfloor (d-1)/2 \rfloor$ where d is the distance of a stabilizer code. An error correction protocol is t-fault tolerant if the following two conditions are satisfied:

1. *ECCP: For any input codeword with an error that can arise from r faults before the protocol and corresponds to the zero cumulative flag vector, if s faults occur during the protocol with $r + s \leq t$, ideally decoding the output state gives the same codeword as ideally decoding the input state.*

2. *ECRP: If s faults occur during the protocol with $s \leq t$, regardless of the number of faults that can cause the input error, the output state differs from any valid codeword by an error that can arise from s faults and corresponds to the zero cumulative flag vector.*

The main difference between these two definitions of FTEC is that Definition 4 considers the *number of faults* that can cause the input (or the output) error instead of the *weight* of the error. An FTEC protocol satisfying Definition 4 can be constructed if we can find syndrome extraction circuits which give a distinguishable fault set (see [26] and the discussion in the next section for more details). It was proved in [26] that a version of threshold theorem proposed in [5] still holds when Definition 1 is replaced by Definition 4. For flag FTEC, using Definition 4 can result in simpler FTEC protocols, so we will use Definition 4 in the protocol development throughout this work.

III.   OPTIMIZATION TOOLS FOR SPACE DECODER

In this work, the term space decoder refers to a process that finds a recovery operator from a given syndrome under the assumption that it is exactly the same as the syndrome of an error that occurred to the codeword. The decoder succeeds if multiplying the error and the recovery operator gives a trivial logical operator (a stabilizer), and it fails if the multiplication gives a nontrivial logical operator. Our goal is to develop a space decoder such that whenever the total number of faults in the whole protocol is $\leq t$, the decoder always succeeds. In this work, we are interested a lookup table-based space decoder for flag FTEC, so the decoder will use both syndrome and flag information obtained during the syndrome measurements. Note that the ability to correct faults for a certain code depends on the structure of the circuits for syndrome extraction such as the ordering of gates.

In this section, we develop optimization tools for space decoder. In Section III A, we discuss how to efficiently construct a lookup table for error decoding for a distinguishable fault set $\mathcal{F}_t$, and introduce the notion of fault code. In Section III B, we discuss the Meet-in-the-Middle technique, an additional technique that can help improving our space decoders for both codes and increase the accuracy of the decoding.

A.   Compact lookup table for minimum weight decoding and fault code

In this section, we discuss how to construct the fault set $\mathcal{F}_t$, verify its distinguishability, and construct the lookup table for error decoding. With our method, we can reduce the memory footprint requirement of the lookup table by 75% for self-dual CSS codes and by 50% for general CSS codes, compared to generic stabilizer codes. We also present the framework of *fault code* that enables fast construction using streamlined Pauli-frame simulation represented as matrix algebra operations over GF(2).

A brief summary of our methods is as follows. Let the *weight of a fault combination* be the number of faults that give rise to the fault combination. The decoding table maps each full syndrome $(\vec{s}, \vec{\mathbf{f}})$ to a recovery operator that corresponds to the combined data error of the minimum-weight fault combination that results in the full syndrome. To construct the decoding table, we start by collecting all weight-1 fault combinations that may arise in the extraction circuits. We map each resulting full syndrome to its corresponding data error. At this point, we say that the *search radius* of the lookup table is 1. Afterwards, we combine pairs of weight-1 fault combinations to create all possible weight-2 fault combinations. The combined data error of each weight-2 fault combination is obtained by simply taking the product of the data errors, and the full syndrome is obtained by adding full syndromes of the weight-1 fault combination modulo 2. If the combining process leads to a new syndrome, we store it in the table. If the process leads to an existing syndrome, we have a collision and do one of the following: (1) If the stored combined data error and the new combined data error are the same up to a stabilizer, then we do nothing. (2) If the stored combined data error and the new combined data error differ by a logical operator (up to a stabilizer), then we raise an error; this implies that $\mathcal{F}_2$ is not distinguishable. At this point, if there is no combination that causes the second case (that is, $\mathcal{F}_2$ is distinguishable), we say that the search radius of the lookup table is 2. We can gradually increase the search radius using similar ideas until we reach the maximum search radius in which the fault set is distinguishable. Here we rely on an efficient representation of the combined data errors using the decomposition proposed in [33].

During sampling, the decoder receives a full syndrome that was measured. When the decoder finds this syndrome in the lookup table, it returns the corresponding *actual recovery operator* (ARO). However, when the decoder cannot find the syndrome in the lookup table, it only returns a so called *canonical recovery operator* (CRO). Each syndrome has a unique canonical recovery operator, which guarantees that applying such an operator to the erroneous encoded state will map it back to the code space, but with a possible logical error.

The full description of our methods is presented below.

1.   Reducing the memory footprint

To decode an $[\![n, k, d]\!]$ stabilizer code, we can construct a lookup table that, for all possible fault combinations of weight 0 to t (where $t = \lfloor \frac{d-1}{2} \rfloor$), stores the full syndrome $\vec{\sigma} = (\vec{s}, \vec{\mathbf{f}})$ as the key and maps the combined data error

as the recovery operator. While this approach works, it is expensive. As the size of the lookup table grows exponentially in n, $n - k$ (the number of generators), and the number of circuit locations, we want to choose a representation such that the data is stored as efficiently as possible. With the generic method, for the full syndrome, $n - k$ bits are required for the syndrome bits and $n - k$ bits for the cumulative flag vector (assuming flag circuits with single flag ancilla for simplicity). Meanwhile, the recovery operator requires $2n$ bits. Thus we have $2(n - k) + 2n$ for each entry in the map. The factor of 2 comes from the symplectic representation of Pauli operators.

Leveraging the structure of our problem, we can significantly improve the size of each entry. As we mainly focus on self-dual CSS codes, the number of used bits can be reduced by half, that is, using $n - k$ bits for the map key and n bits for the recovery operator are sufficient. Moreover, we can further reduce the number of bits for the recovery operator to k using the following two key ideas:

1. In general, for an $[\![n, k, d]\!]$ code, each Pauli operator $P \in \mathcal{P}_n$ can be decomposed as a product $P = EML$ of a pure error E, a stabilizer $M \in \mathcal{S}$, and a logical operator $L \in \overline{\mathcal{P}}_k$ (where $\overline{\mathcal{P}}_k$ is the k-dimensional logical Pauli group) [33]. We define a fixed set of pure errors called *canonical recovery operators* (CRO), one CRO for each unique syndrome $\vec{s}$.

2. Given a syndrome $\vec{s}(E)$, the goal of decoding is to find a recovery operator R such that $RE \in \mathcal{S}$, thus R. For any possible Pauli error, we only have to store its *logical class*, a value that indicates how the error is related to a CRO with the same syndrome. This enables the map value to be only $2k$ bits of information in general and k bits in case that the code is a self-dual CSS code. In this latter case and with $k = 1$, the logical class is 0 if multiplication of the Pauli operator and the CRO with the same syndrome is in the stabilizer group, otherwise the logical class is 1.

Altogether, we gain a $1 - \frac{n-k+k}{2(n-k)+2n} = 1 - \frac{1}{4-2k/n} \approx 75\%$ reduction in the size of the table (assuming $n \gg k$). For a CSS code which is not self-dual, the gain is smaller but still a significant 50%.

2. Constructing the lookup table

We now explicitly describe an algorithm to construct the lookup table. During the construction of the lookup table, we have a systemic way to enumerate fault combinations with their full syndromes and combined data errors instead of running through a circuit simulator for each case. The exhaustive enumeration of all possible fault combinations of weight 0 to t is done in two steps. First, we enumerate the single faults and capture their full syndrome and logical class in a single column of the *fault check matrix* using matrix algebra over GF(2) to represent the propagation of errors in our syndrome extraction circuits. Then, we combine these columns in all possible combinations of 0 to t faults ($\sum_{i=0}^{t} \binom{n}{i}$ combinations in total), while keeping track of the weight of each fault combination. This last step both verifies whether $\mathcal{F}_t$ is distinguishable (which is equivalent to verifying whether the protocol is distance preserving), and builds a lookup table for the decoder.

Enumerating weight-1 faults— From here on we will only consider a self-dual CSS code. To list all possible single faults under the circuit-level depolarizing noise model, it is sufficient to consider all possible weight-1 faults within a single round of syndrome measurements. Each column of the fault check matrix H_{f} describes for each possible weight-1 fault what its full syndrome and its logical class are. As the logical class of each single fault depends on how its CRO is defined, we denote the fault check matrix in which the CROs are defined using a right inverse H^{-1} of H by $H_{\mathrm{f}}|_{H^{-1}}$. The high-level structure of H_{f} consists of three major groups of rows and three major groups of columns. The three groups of rows are the $(n - k)/2$ generator bits, the $(n - k)/2$ flag bits, and the k bits for the logical class. Each single fault which is represented by a column of $\bar{H}_{\mathrm{f}}$ can be put into one of the following three categories:

1. **Pure data qubit errors** result in generator bits that can be characterized with the parity check matrix H of the CSS code. They do not trigger flags, resulting in all-zero flag bits. Given a right inverse H^{-1} of H ($HH^{-1} = I_{(n-k)/2}$), the CRO R of each pure data qubit error E can be described by each column of $H^{-1}H$ (since the syndromes of CROs are $H(H^{-1}H) = (HH^{-1})H = H$), thus the product RE of each E can be described by each column of $I_n \oplus H^{-1}H$ (where the matrix addition, denoted by $\oplus$, and multiplication are over GF(2)). If E is an X-type (or a Z-type) error, the logical class of RE is described by a k-bit string in which the i-th bit indicates whether RE anticommutes with $\bar{Z}_i$ (or $\bar{X}_i$). That is, the logical classes of all pure data qubit errors are described by

$$\begin{pmatrix} J_1^T(I_n \oplus H^{-1}H) \\ J_2^T(I_n \oplus H^{-1}H) \\ \cdots \\ J_k^T(I_n \oplus H^{-1}H) \end{pmatrix},$$

where J_i is the column vector representing $\bar{Z}_i$ (or $\bar{X}_i$).

2. **Flag ancilla preparation or measurement errors** do not propagate to data qubits, thus each single-flag error will result in a single flag bit. Therefore, all errors of this type have the all-zero syndrome and logical class 0, while the flag bits can be easily represented by the $(n - k)/2 \times (n - k)/2$ identity matrix.

3. **Gate faults** that cause errors on the syndrome ancilla which can propagate to data and flag qubits — we order these faults by top-down and left-right place of occurrence and capture their effect in syndrome bits, flag bits, and logical class. The part of the effective matrix corresponding to this type of faults is denoted by $H_{\mathrm{f,gate}}$.

Note that single measurement and reset errors on the syndrome ancilla are ignored during this analysis as their effects would be removed by the time decoder through the repetition of syndrome measurements.

In case that n is odd, $k = 1$, and logical X and logical Z operators are transversal, the fault check matrix is,

$$H_{\mathrm{f}}|_{H^{-1}} = \left(\begin{array}{c} H \\ 0 \\ J^T(I_n \oplus H^{-1}H) \end{array} \middle| \begin{array}{c} 0 \\ I_{(n-1)/2} \\ 0 \end{array} \middle| H_{\mathrm{f,gate}} \right),$$

where J is the all-one column vector of length n (representing $X^{\otimes n}$ or $Z^{\otimes n}$).

As an example, consider the first group of columns for the $[\![7,1,3]\!]$ Steane code [12] whose stabilizer generators can be defined by the parity check matrix,

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

One can pick its right inverse H^{-1} as follows:

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

We can see that each column of H^{-1} gives a Pauli operator for each syndrome bit. For a data error E of any weight, the syndrome $\vec{s}(E) = HE$ can be recovered with the CRO defined by $R(\vec{s}(E)) \equiv H^{-1}\vec{s}(E)$ as

$$\begin{aligned} \vec{s}(E \oplus R) &= H(E \oplus R) \\ &= HE \oplus HH^{-1}HE \\ &= HE \oplus HE = 0. \end{aligned}$$

For example, for $E = (0110000)^T$, $\vec{s}(E) = (001)^T$ and $R(\vec{s}(E)) = (1000000)^T$, thus $RE = (1110000)^T$, for which the syndrome is trivial (as RE is a logical operator).

For errors of weight 1 on the data qubits, the operator RE of each error can be represented by each column of $I_n \oplus H^{-1}H$. Since the logical class of RE can be determined by its weight parity, the logical classes of this type of errors are the row of $L \equiv J^T(I_n \oplus H^{-1}H)$ where J is the all-one column vector. That is for the Steane code,
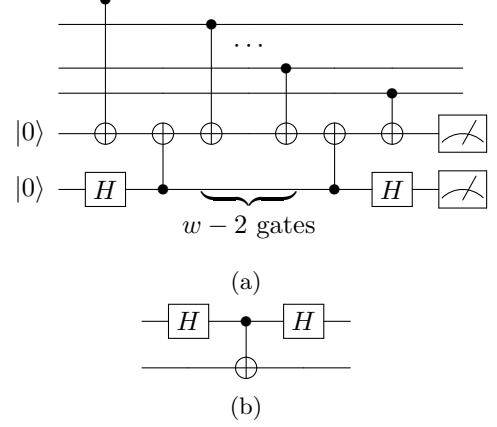

(a)

(b)

FIGURE 4: (a) A flag circuit for measuring a Z-type stabilizer generator of weight w in this work. A flag circuit for measuring a X-type stabilizer generator of weight w can be obtained by replacing each CNOT gate that connects the data qubit to the syndrome ancilla with the gate in (b).

the part of $H_{\mathrm{f}}|_{H^{-1}}$ corresponding to pure data qubit errors is

$$\left(\begin{array}{c} \hline H \\ \hline 0 \\ \hline J^T(I_n \oplus H^{-1}H) \end{array} \right) = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

Constructing $H_{f,gate}$ — In this work, we focus on the case that any Z-type or X-type stabilizer generator of weight w is measured using a flag circuit with a single flag ancilla similar to the circuit in Fig. 4 (with a slight modification, similar construction for a general flag circuit can also be made). For $H_{\mathrm{f,gate}}$, we are interested in how the errors propagate from the syndrome ancilla to the data qubits and the flag ancilla. Here we generalize the representation of a syndrome extraction circuit by a binary matrix proposed in [34, 35]. Given single-flag syndrome extraction circuits for all stabilizer generators and the CNOT ordering for each circuit, $H_{\mathrm{f,gate}}$ can be calculated via the *propagator matrix P* and the *aggregator matrix A*, defined as follows: For the error correction protocol with n data qubits and r flag bits (which is the same number as the number of X or Z stabilizer generators), The matrix P has $n + r$ rows. The number of columns of P is $\sum_{i=1}^{r}(w(g_i)+2)$, where $w(g_i)$ is the Hamming weight of the i-th stabilizer generator. This is from the fact that for each CNOT gate in the single-flag syndrome extraction circuits, the only fault that can lead to a unique data error after propagation is the fault that leads to a single Z error on the target qubit of the CNOT (which

is the syndrome ancilla). To simplify the construction, we construct a submatrix P_i of size $(n + r) \times (w(g_i) + 2)$ for each row g_i of H (i.e each stabilizer generator), then concatenate the submatrices to get,

$$P = (P_1 P_2 \ldots P_r). \qquad (3)$$

As the order of the CNOT gates matters in subtle ways, for a given stabilizer generator g_i, we represent the CNOT ordering by the permutation π_i : $\{1, 2, \ldots, w(g_i)\} \rightarrow supp(g_i)$, where $\pi_i(j)$ indicates the control (data) qubit of the j-th CNOT (the target qubit is always the syndrome ancilla). π_i can also be represented by a list. For example, two possible permutations of CNOT gates in the syndrome extraction circuit for measuring g_1 of the $[\![7, 1, 3]\!]$ Steane code are $\pi_1 = [4, 5, 6, 7]$ and $\pi_1 = [4, 6, 5, 7]$.

To construct P_i, we iterate from $j = 1$ to $w(g_i)$, and create a column for each iteration with all zeros except for the 1 in row $\pi_i(j)$. We then insert an all-zero column on the second from the left and the second from the right positions (which represent the flag CNOTs), and set its value to 1 at row $n + i$. In our running example of $g_1 = (0001111)$, for a permutation of $\pi_1 = [4, 6, 5, 7]$,

$$P_i = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \qquad (4)$$

The aggregator matrix A plays the role of propagating the errors to the end of the syndrome measurement circuits. For each g_i, we define A_i to be a square matrix of size $(w(g_i) + 2) \times (w(g_i) + 2)$ having a lower triangle set to all 1s, and define $A = \bigoplus_{i=1}^{r} A_i$ to be the direct sum of all A_i's. In our example case of g_1,

$$A_i = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \qquad (5)$$

Multiplying the propagator and the aggregator matrices yields,

$$PA = \begin{pmatrix} F \\ \hline \Phi \end{pmatrix} = \begin{pmatrix} F_1 & F_2 & \ldots & F_r \\ \hline \Phi_1 & \Phi_2 & \ldots & \Phi_r \end{pmatrix}, \qquad (6)$$

where columns of the submatrices F_i are the final Pauli operators, and columns of the submatrices Φ_i are the cumulative flag vectors after measuring g_i and having a fault propagated from the syndrome ancilla to the data qubits at the location corresponding to the given column.

Next, we find the syndromes for these Pauli operators by multiplying them with the parity check matrix,

$$S = HF. \qquad (7)$$

Then, for each syndrome, we define the CRO based on the right inverse H^{-1},

$$\Theta = H^{-1}S. \qquad (8)$$

Finally, we determine the logical class L for each of the faults by adding the parity of the CRO and the propagated, final data error,

$$L = J^T(F \oplus \Theta). \qquad (9)$$

As a result, the part of $H_{\mathrm{f}}|_{H^{-1}}$ corresponding to the gate faults is,

$$H_{\mathrm{f,gate}} = \begin{pmatrix} S \\ \hline \Phi \\ \hline L \end{pmatrix} = \begin{pmatrix} HF \\ \hline \Phi \\ \hline J^T(I_n \oplus H^{-1}H)F \end{pmatrix}. \qquad (10)$$

The relationship between the full syndrome, the data error, and the CRO of each fault is as follows: Suppose that the i-th column of $H_{\mathrm{f}}|_{H^{-1}}$ (which represents a single fault on the i-th location) contains error syndrome $\vec{s}_i$, flag vector $\vec{f}_i$, and logical class l_i. The CRO of the fault is $H^{-1}\vec{s}_i$, while the data error of the fault is $l_i J \oplus H^{-1}\vec{s}_i$. That is, in case of a single fault, the *actual recovery operator* (ARO) we need to apply when finding the full syndrome $(\vec{s}_i, \vec{f}_i)$ is $l_i J \oplus H^{-1}\vec{s}_i$.

Verifying distinguishability and building the lookup table— The fault set $\mathcal{F}_t$ is distinguishable if and only if there is no fault combination from up to $2t$ faults that gives a non-trivial logical operator with trivial full syndrome (see Proposition 1 in [26]). As the fault check matrix already contains all the possible single faults, in case of $t = 1$, we only need to extend the matrix by a column with all zeros (which represents 0 faults) and check whether there is a pair of columns which are the same except for the logical class. If there is, the combined data errors of one or two faults add up to an undetectable logical operator, meaning that $\mathcal{F}_1$ is not distinguishable.

When $t \geq 2$, we populate the cache with the logical classes of higher-weight fault combinations by simply combining all possible fault combinations of lower-weight fault combinations while keeping track of the weights of the fault combinations. We describe the i-th fault combination as a key-value pair $[(\vec{s}_i, \vec{f}_i) : (l_i, w_i)]$ where $(\vec{s}_i, \vec{f}_i)$ is the full syndrome, l_i is the logical class, and w_i is the weight of the fault combination. Combining the i-th and the j-th fault combinations gives

$[(\vec{s}_i \oplus \vec{s}_j, \vec{f}_i \oplus \vec{f}_j) : (l_i \oplus l_j, w_i + w_j)]$. As we aim to check whether $\mathcal{F}_t$ is distinguishable, we fill up the cache by combining any pair of fault combinations that satisfy $w_i + w_j \le t$. In case that the process gives the new key (the full syndrome) that already exists in the cache, we have a key conflict. This can be one of the following cases:

1. The new and the existing fault combinations have the same full syndrome and the same logical class but have different weights. In this case, we store the fault combination with smaller weight in the cache.

2. The new and the existing fault combinations have the same full syndrome but have different logical classes. As the sum of weights of these two fault combinations is $\le 2t$, we raise an error—there exists a fault combination from up to $2t$ faults that gives a non-trivial logical operator with trivial full syndrome, that is, $\mathcal{F}_t$ is not distinguishable.

If at the end we find that $\mathcal{F}_t$ is distinguishable, we can construct a lookup table of search radius t from the cache as follows: for each key-value pair $[(\vec{s}_i, \vec{f}_i) : (l_i, w_i)]$ in the cache, we store a new key-value pair $[(\vec{s}_i, \vec{f}_i) : l_i J \oplus H^{-1} \vec{s}_i]$ in the lookup table (the weights are not necessary for decoding, though it might be useful for estimating the number of faults that causes the full syndrome). That is, $l_i J \oplus H^{-1} \vec{s}_i$ is the ARO for the full syndrome $(\vec{s}_i, \vec{f}_i)$. When perform error decoding, the ARO is applied if the full syndrome obtained from measurements is found on the lookup table; otherwise, the CRO $(H^{-1}\vec{s}_i)$ is applied.

The lookup table then can be stored in an efficient binary format as needed on disk or memory. In Table II we displayed the metrics related to the lookup table decoder obtained by the algorithm above.

In summary, we perform an exhaustive search of fault combinations which gives us a lookup table with search radius t; this is equivalent to verifying the distinguishability of $\mathcal{F}_t$. If we can construct the lookup table with $t = \tau = \lfloor (d-1)/2 \rfloor$, we have a minimum-weight decoder that is distance-preserving under the circuit-level depolarizing noise model. As a hash-table requires $\mathcal{O}(1)$ amortized complexity for lookup, this decoder is also relatively fast for numerical simulations or real-time decoding compared to more complicated algorithms such as MaxSAT-decoding [36], neural-network-based decoder [37], or the restriction decoder [38] with minimum weight perfect matching decoding [39], all of which have at least $\mathcal{O}(n)$ complexity. However, the table size scales exponentially in the number of qubits, locations, and stabilizer generators, and thus constructing the lookup table may be impractical for a code of high distance.

3. The fault code

Any CSS code can be defined by its parity check matrix H which maps a bitstring representing a combination of *errors* on the data qubits to the error syndrome of the error combination. In the case of flag FTEC where the circuit-level noise model is considered, we can use similar ideas and define a *fault code* by the fault check matrix H_f which maps a bitstring representing a combination of *possible faults* to the full syndrome of the fault combination (which includes the error syndrome of the combined data error and cumulative flag vector) and the logical class relative to the CRO for the syndrome. It should be noted that the distance of the fault code might be lower than the distance of the underlying CSS code; this depends on the syndrome extraction circuits which affect the distinguishability of the fault set. We can define the *effective distance* d_eff to be the minimum number of faults that can give a fault combination with a non-trivial logical operator and the trivial full syndrome. The number of faults t_eff that the fault code can correct is $t_\mathrm{eff} = \lfloor (d_\mathrm{eff} - 1)/2 \rfloor$ (this is the maximum number of t in which $\mathcal{F}_t$ is distinguishable). If the effective distance and the code distance are equal, we say that the error correction protocol is distance preserving. Calculating the distance of classical codes can be done by determining the spark of the parity check matrix H, which is known to be NP-hard, in general [40]. However, the spark algorithm does not work in the case of degenerate CSS codes, as it reports only the minimum weight of the stabilizers which is a lower bound on the code distance [32]. Our algorithm described in this section can be viewed as a modified spark algorithm that uses the logical class information to calculate the distance of the code (based on H) and also the effective distance of the fault code (based on H_f).

The perspective of the fault code can be also useful to extend a technique frequently used for error sampling (in qecsim [41], for example) to the circuit-level noise model beyond the code capacity noise model (memory errors only) and phenomenological noise model (both memory and measurement errors). Here, a randomly-generated column vector of Hamming weight w now represents faults on w locations instead of errors on w qubits. Suppose that the vector $\vec{v}$ represents the fault combination and $H_\mathrm{f}\vec{v}$ gives the full syndrome $(\vec{s}(\vec{v}), \vec{f}(\vec{v}))$ and the logical class $l(\vec{v})$. In an error correction simulation, the decoder can predict the recovery operator $\vec{r}$ based on the full syndrome. We will find that the predicted recovery operator causes a logical error if and only if $l(\vec{v})$ and $l(\vec{r})$ differ. In principle, this method can lead to a better sampling rate compared to running the full circuit simulation for each sample. However, one needs to be aware of the probability distribution when generating vectors representing the fault combinations as each possible single fault might not occur at the same rate.

B. Meet-in-the-Middle Technique

If the fault set $\mathcal{F}_t$ of each code is distinguishable, the flag FTEC protocol can correct up to t faults with certainty. However, whenever $t + 1$ or more faults occur, the error correction is not guaranteed; our decoder can either remove the error or cause a logical error on the encoded state. Although the probability of having $t + 1$ or more faults is $\mathcal{O}(p^{t+1})$, being able to correct more cases of faults can lead to a higher pseudothreshold. In this section, we introduce the Meet-in-the-Middle technique which can help correct errors in case there are more than t faults in our FTEC protocol. Note that this technique is general and it could help any FTEC protocol with a table-based decoder to correct faults more than its capability if the stabilizer code being used is not a perfect (or a perfect CSS) code.

The Meet-in-the-middle (MIM) technique is inspired by the bidirectional search algorithm [42] to improve the table-based decoder previously discussed in Section II B (see also Section III A) in case that the decoder cannot find in its lookup table the full syndrome obtained from measurements. Consider the case that the fault set $\mathcal{F}_t$ is distinguishable and a lookup table of search radius t can be constructed. Suppose that more than t faults occur and the full syndrome is $(\vec{s}_m, \vec{\mathbf{f}}_m)$ which is not in the lookup table. The table-based decoder discussed in Section II B will return the canonical recovery operator that may cause a logical error after correction. To make successful error correction in such cases more probable, one could in principle construct a lookup table with a search radius larger than t by relaxing the distinguishability requirement for fault combinations with weights higher than t. However, this can be impractical as the number of fault combinations grows too fast when the search radius increases.

To overcome this issue, we instead conduct a *search during decoding* starting from the missing syndrome $(\vec{s}_m, \vec{\mathbf{f}}_m)$. That is, we construct another decoding table, called the MIM table, with search radius at most $\rho \le t$ using ideas similar to the original lookup table, but we also add $(\vec{s}_m, \vec{\mathbf{f}}_m)$ to the map key before storing the syndrome in the MIM table and check whether it is in the decoding lookup table or not. If a new map key in the MIM table is the same as some map key in the decoding lookup table, the search stops and the decoder constructs a recovery operator from two combined data errors from the MIM table and the decoding table that correspond to the two map keys. If the MIM search radius reaches ρ and no matching syndrome is found, the decoder returns the CRO for the full syndrome. Using the recovery operator obtained from this method, we can correct up to $t + \rho$ faults with probability higher than using the CRO of the full syndrome only.

An example of error decoding using a lookup table and the MIM technique is illustrated in Fig. 5. In our FTEC protocols for hexagonal color codes of distance 3, 5, 7, and 9, we find that constructing the MIM table with

search radius $\rho = t$ is sufficiently fast to be used at runtime. Note that the MIM technique does not guarantee successful error correction due to potential degeneracy in syndromes above the guaranteed number of correctable faults. However, we do find numerically that the MIM technique has a positive impact on the performance of our decoders for the hexagonal color codes for distance 3, 5, 7, and 9.

IV. OPTIMIZATION TOOLS FOR TIME DECODER

In general, faults can happen at any point during syndrome measurements, and the syndrome obtained at each round of measurements may not be the *correct syndrome* (the syndrome of the combined data error at the end of that round). In particular, measurement errors can lead to a syndrome that differs from the correct syndrome by some bits. Errors on data or ancilla qubits that happen the middle of the syndrome extraction can also result in a syndrome that only captures some parts of the correct syndrome. Applying a space decoder to a faulty syndrome can lead to an incorrect recovery operation. For this reason, multiple rounds of syndrome measurements must be performed.

The goal of a time decoder is to find a round with a correct syndrome at least at one point in the whole syndrome measurement process. If this can be done, an FTEC protocol satisfying both conditions in Definition 4 can be constructed. Note that according to the definition, it is sufficient to consider only the case that the total number of faults in the whole protocol is no more than t, where t is the number of errors that a stabilizer code being used can correct. This is because the failure probability of the FTEC protocol (the probability of having $t + 1$ or more faults in the protocol) will be $\mathcal{O}(p^{t+1})$ similar to the failure probability of an ideal error correction with the same stabilizer code. (Nevertheless, in terms of better decoding accuracy, it is beneficial to consider correcting some cases of $t + 1$ or more faults as suggested by the MIM technique in Section III B.)

In this section, we develop several types of time decoders for flag FTEC, building on the ideas of adaptive decoders in [29]. Different time decoders use different fault count estimation procedures. The decoders presented in this paper are tailored to a CSS code, but they can be generalized for any stabilizer code. We also provide extended techniques which can further improve our flag FTEC protocols when the CSS code being used is self-dual.

A. Shor time decoder for flag FTEC

In Shor's original approach [1], the syndrome extraction is repeated until the same syndrome appears $t + 1$ times in a row. Observe that for R repeated but untrust-
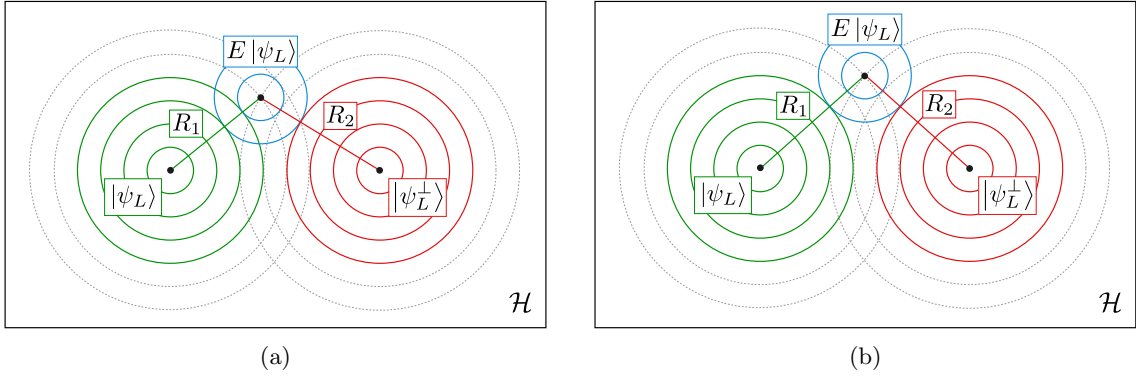
FIGURE 5: An illustration of the error decoding using a lookup table and the MIM technique on the Hilbert space $\mathcal{H} = \mathbb{C}^{\otimes n}$ of the physical qubits. A code of distance 9 is considered in this example. Using a lookup table with search radius 4 only, any erroneous states lying on the green (or red) circles, which are up to 4 faults away from the logical state $|\psi_L\rangle$ (or $|\psi_L^\perp\rangle$), will be recovered to the logical state $|\psi_L\rangle$ (or $|\psi_L^\perp\rangle$). Consider the erroneous state $E|\psi_L\rangle$ which is not on any green or any red circle. In (a), $E|\psi_L\rangle$ is 5 faults away from $|\psi_L\rangle$ and 6 faults away from $|\psi_L^\perp\rangle$. Using the MIM table of radius 1, the recovery operator found by the decoder is R_1. Since $R_1 E$ is a stabilizer, R_1 brings the state back to the original state $|\psi_L\rangle$. In (b), $E|\psi_L\rangle$ is 6 faults away from both $|\psi_L\rangle$ and $|\psi_L^\perp\rangle$. Using the MIM table of radius 2, the recovery operator found by the decoder is either R_1 such that $R_1 E$ is a stabilizer, or R_2 such that $R_2 E$ is a nontrivial logical operator. In this case, the state after recovery can be either $|\psi_L\rangle$ or $|\psi_L^\perp\rangle$.

worthy syndromes, at least R faults are required to make them the same (we can think of having exactly the same measurement errors for example). Therefore, to make sure that a round with a correct syndrome exists when considering the case with up to t faults, it is sufficient to wait for $t + 1$ repeated measurements. A time decoder with this stop condition will be referred to as *Shor time decoder*.

It is possible to rephrase the Shor time decoder using the notion of difference vector. For a syndrome history $(\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_m)$ of length m, we define a *difference vector* $\vec{\delta}$ to be an $(m-1)$-bit string in which δ_i is 0 if $\vec{s}_{i+1} = \vec{s}_i$, or δ_i is 1 if $\vec{s}_{i+1} \neq \vec{s}_i$. As two repeated syndrome measurements are represented by a zero in the difference vector, Shor's method can be reformulated as waiting for t consecutive zeros in $\vec{\delta}$.

As we aim to correct no more than t faults, the analysis of our time decoders can be made easier by thinking about the budget of t faults. Shor's method spends all of this budget on counting consecutive zeros in the difference vector and is completely oblivious to other parts of the syndrome history (because the counter is reset whenever bit one appears). We call the parts of the syndrome history outside of the zero substring the *context* of the zero substring. As Shor's method does not take the context into account, we call this strategy "context-unaware". In the worst-case scenario for the Shor time decoder, $(t+1)^2$ rounds of syndrome measurements are done before the stopping condition is satisfied. The context of the zero substrings contains useful information and not counting the faults in the context results in underestimating the number of faults that can cause a given syndrome history. Context-aware strategies that have a better estimate of the number of faults can stop earlier, and execute fewer measurements, resulting in higher

pseudothresholds.

As flag circuits are used in the syndrome extraction, we also obtain a flag vector history $(\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_m)$ from m rounds of syndrome measurements, which also leads to a cumulative flag vector history $(\vec{\mathbf{f}}_1, \vec{\mathbf{f}}_2, \ldots, \vec{\mathbf{f}}_m)$. Note that the calculation of a difference vector *does not* involve flag vectors; since the cases of faulty flag qubit measurements are considered when we evaluate the distinguishability of a fault set, all flag measurement outcomes are considered correct and can be used for error decoding. Our goal is to find a round such that all syndrome bits are correct.

The correct syndrome will be used in conjunction with the flag information obtained *right before* the measurements of the correct syndrome. Suppose that the code being used is a CSS code, and X-type generator measurements at round i (which lead to $\vec{s}_{i,x}$, $\vec{f}_{i,x}$, and $\vec{\mathbf{f}}_{i,x}$) are done before Z-type generator measurements (which leads to $\vec{s}_{i,z}$, $\vec{f}_{i,z}$, and $\vec{\mathbf{f}}_{i,z}$). If the syndrome from round l is correct according to Shor time decoder, Z-type (or X-type) error correction will be done using $\vec{s}_{l,x}$ and $\vec{\mathbf{f}}_{l-1,z}$ (or $\vec{s}_{l,z}$ and $\vec{\mathbf{f}}_{l,x}$). We also use similar ideas for error correction with other time decoders.

Suppose that a table-based space decoder for flag FTEC can be constructed (as discussed in Section III), a flag FTEC protocol with Shor time decoder is as follows:

Protocol 1. *Flag FTEC protocol with Shor time decoder*
   Let $t = \lfloor (d-1)/2 \rfloor$ be the number of errors that a stabilizer code of distance d can correct. Let $\vec{s}_i = (\vec{s}_{i,x}, \vec{s}_{i,z})$ and $\vec{f}_i = (\vec{f}_{i,x}, \vec{f}_{i,z})$ be syndrome and flag vector obtained from the i-th round of full syndrome measurements with flag circuits. Let the cumulative flag vector at the i-th round be $\vec{\mathbf{f}}_i = (\vec{\mathbf{f}}_{i,x}, \vec{\mathbf{f}}_{i,z}) = \sum_{j=1}^{i} \vec{f}_j \pmod 2$. After the i-th round with $i \geq 2$, calculate δ_{i-1}. Repeat syndrome measurements until the last t bits of $\vec{\delta}$ is zero or the total

number of rounds reaches $(t+1)^2$. *Suppose that the latest round is round* l. *Perform* Z-*type error correction using* $(\vec{s}_{l,x}, \vec{\mathbf{f}}_{l-1,z})$, *and perform* X-*type error correction using* $(\vec{s}_{l,z}, \vec{\mathbf{f}}_{l,x})$.

B. Adaptive time decoder for flag FTEC

Recently, FTEC protocols with adaptive syndrome measurement techniques have been proposed in [29]. In their work, each stabilizer generator is measured using a syndrome extraction circuit with a cat state (similar to the circuit originally proposed in [1]). The authors show that by using context-aware strategies, it is possible to reduce the number of syndrome measurement rounds in the worst-case scenario from $(t+1)^2$ rounds to $(t+3)^2/4 - 1$ rounds. The FTEC protocol with adaptive strong decoder in [29] satisfies the FTEC conditions in Definition 1 and is applicable to any stabilizer code. In this work, we extend the adaptive measurement techniques to flag FTEC and develop flag FTEC protocols that satisfies the revised FTEC conditions in Definition 4.

1. Counting faults in correlated and uncorrelated bit histories

Let us first consider a way to estimate the number of occurred faults from a given difference vector $\vec{\delta}$ as proposed in [29]. A single fault can cause either one or two consecutive bits of ones in $\vec{\delta}$ (see Type I and Type II faults in [29]). Thus, for each substring $\vec{\kappa}$ in $\vec{\delta}$, the number of faults that can cause such a substring is bounded from below by the number of 11 sequences plus the number of remaining 1s in $\vec{\kappa}$.

Suppose that the difference vector is of the form $\vec{\delta} = \eta_1 1 \eta_2 1 \ldots 1 \eta_c$. For each η_j of length $\gamma_j \geq 1$ (where $2 \leq j \leq c-1$), we define α to be the total number of non-overlapping 11 sequences plus the total number of remaining 1s before the substring $1\eta_j 1$, and define β_j similarly but for the substring after $1\eta_j 1$ (for η_1 and η_c, β_1 and α_c are defined similarly to those of other η_j's, and we let $\alpha_1 = 0$, $\beta_c = 0$). η_j of length γ_j corresponds $\gamma_j + 1$ consecutive rounds with the same syndrome, so the number of rounds that can cause these rounds to give incorrect syndromes is at least $\gamma_j + 1$. Therefore, under the assumption that there are at most t faults in the whole protocol, if we find that there exists γ_j such that $t - \alpha_j - \beta_j < \gamma_j + 1$, the syndromes of the $\gamma_j + 1$ rounds that give rise to η_j cannot be all incorrect. For this reason, at least one syndrome corresponding to η_j is correct and can be used for error correction (see the full analysis in [29] for more details).

There are multiple, increasing fine-grained ways of estimating the number of faults in the context around each zero substring in the difference vector. Here we use the term *bit history* as a general term for a series of syndrome bits (measurement outcomes) from a given stabilizer generator, a given flag bit (the measurement outcome of a flag qubit), or bits in a difference vector (that is taken as the difference history of a group of bits). A key element in this discussion is the notion of correlated and uncorrelated bit histories.

Under the assumed error model, two-bit histories are uncorrelated if they are independent of each other. For example, in our case, the circuit-level depolarizing channel is memoryless and each fault can cause either one or two consecutive bits of ones, thus different sections of the same syndrome bit history that are at least two bits apart are uncorrelated as they are independent in time. Similarly, in space, if there are no shared qubits between two generators, their syndrome bit histories are completely independent. Also, flag qubits are always reset between rounds of measurements, and thus all flag bits are completely independent. However, when two stabilizer generators share at least one qubit, their syndrome bit histories are correlated. Similarly, due to hook errors, the flag qubit's bit history and the syndrome bit history of that same stabilizer generator are correlated.

Our goal is to estimate the number of faults that occurred from a given bit history in the case of flag FTEC. Estimates from uncorrelated histories can be summed together. When two or more estimates are from correlated histories, the best we can do is to take the maximum of those estimates. Note that the total estimates must not exceed the actual total number of occurred faults in any case, otherwise, the error correction protocol will not be fault tolerant.

For the estimation in [29] which is discussed earlier, the bits of the syndrome history before and after each substring $1\eta_j 1$ are uncorrelated to the bits within η_j under the memoryless depolarizing channel assumption. This means that α_j and β_j, which are the minimum numbers of faults that can cause the substring before and after $1\eta_j 1$, can be independently estimated. The estimate of the number of faults for the context outside of the zero substring η_j is, therefore, $\alpha_j + \beta_j$.

In this work, we further extend the fault counting idea to flag FTEC in which flag circuits with single flag qubit are used for syndrome extraction. Below, we will discuss two types of adaptive time decoders with different stop conditions, namely *one-tailed and two-tailed adaptive time decoders*. Both protocols are applicable to any stabilizer code as long as flag circuits for the code that give a distinguishable fault set can be found. The flag FTEC protocol with one-tailed adaptive time decoder satisfies the FTEC conditions in Definition 4, thus it is applicable to any fault-tolerant quantum computation as long as the fault-tolerant implementation of other operations (gate, state preparation, or measurement) also satisfies the revised definition of fault tolerance in [26]. Meanwhile, the flag FTEC protocol with two-tailed adaptive time decoder does not satisfy the FTEC conditions in Definition 4 as the output error may correspond to a nontrivial cumulative flag vector, hence it is only appli-

cable to quantum memory. Nevertheless, for a self-dual CSS code, the FTEC protocol with the two-tailed adaptive time decoder can be applied to any fault-tolerant Clifford computation if the cumulative flag vector is processed appropriately. An analysis of this extension will be discussed in Section IV C.

2. Two-tailed adaptive time decoder

For the substring $1\eta_j 1$ in $\vec{\delta}$, suppose that bit one on the left of η_j is the i_1-th bit of $\vec{\delta}$, and bit one on the right of η_j is the i_2-th bit of $\vec{\delta}$. Let $\alpha_j, \beta_j, \gamma_j$ be defined as before, and let μ_j, ν_j be the total numbers of nonzero flag bits obtained from round 1 to round i_1 and from round $i_2 + 1$ onward. Also, let ω_j be the sum of the numbers of flag bits that *exceed 1 bit per round* during round $i_1 + 1$ to round i_2. For example, consider the substring $1\eta_j 1 = 1001$ in the example below:

$$\text{Round: } 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10$$
$$\#\text{ flag bits: } 1\ 0\ 2\ 0|0\ 2\ 1|0\ 0\ 1$$
$$\vec{\delta}:\ 1\ 1\ 0|1\ 0\ 0\ 1|0\ 1$$

In this example, $\alpha_j = 1$, $\beta_j = 1$, $\gamma_j = 2$, $\mu_j = 3$, and $\nu_j = 1$, and $\omega_j = 1$.

Since a single fault can cause both nontrivial flag bits and syndrome differences (that is, syndrome bits and flag bits are correlated), one has to make sure that the number of faults is not overcounted. The numbers of faults that can cause bit histories before and after $1\eta_j 1$ are bounded from below by $\tilde{\alpha}_j = \max(\alpha_j, \mu_j)$ and $\tilde{\beta}_j = \max(\beta_j, \nu_j)$, respectively. So an estimate of the number of faults for the context outside of η_j is $\tilde{\alpha}_j + \tilde{\beta}_j$.

Next, let us consider η_j of length γ_j which corresponds $\gamma_j + 1$ consecutive rounds with the same syndrome. To make all syndromes in this region incorrect, it requires at least one fault per round. So if we find a round with more than one flag bit, the number of flag bits that exceed one bit per round can be a part of the total estimate. That is, for each η_j, the total estimate is $\tilde{\alpha}_j + \tilde{\beta}_j + \omega_j$.

Under the assumption that there are at most t faults in the whole protocol, if we find that there exists γ_j such that $t - \tilde{\alpha}_j - \tilde{\beta}_j - \omega_j < \gamma_j + 1$ (or equivalently, $\tilde{\alpha}_j + \tilde{\beta}_j + \gamma_j + \omega_j \geq t$), we know that a syndrome of at least one round in the $\gamma_j + 1$ rounds that give rise to η_j must be correct.

Another way to find a correct syndrome is to estimate the total number of faults that can cause the whole syndrome and flag bit histories. Let N_{11} be the total number of non-overlapping 11 sequences in the whole $\vec{\delta}$. Assuming that there are at most t faults in the whole protocol, if $N_{11} \geq t$, the last round must have a correct syndrome.

Suppose that a table-based space decoder for flag FTEC can be constructed, a flag FTEC protocol with two-tailed adaptive time decoder is as follows:

Protocol 2. *Flag FTEC protocol with two-tailed adaptive time decoder*

Let $t = \lfloor (d-1)/2 \rfloor$ be the number of errors that a stabilizer code of distance d can correct. Let $\vec{s}_i = (\vec{s}_{i,x}, \vec{s}_{i,z})$ and $\vec{f}_i = (\vec{f}_{i,x}, \vec{f}_{i,z})$ be syndrome and cumulative flag vector obtained from the i-th round of full syndrome measurements with flag circuits. After the i-th round with $i \geq 2$, calculate δ_{i-1}. Repeat syndrome measurements until one of the following conditions is satisfied, then perform error correction using the error syndrome corresponding to each condition:

1. *For each η_j in $\vec{\delta}$, calculate $\tilde{\alpha}_j, \tilde{\beta}_j, \gamma_j, \omega_j$. If at least one η_j with $\tilde{\alpha}_j + \tilde{\beta}_j + \gamma_j + \omega_j \geq t$ is found, stop the syndrome measurements. Let l be the last round of the $\gamma_j + 1$ rounds that correspond to η_j. Perform Z-type error correction using $(\vec{s}_{l,x}, \vec{f}_{l-1,z})$, and perform X-type error correction using $(\vec{s}_{l,z}, \vec{f}_{l,x})$.*

2. *Calculate N_{11} from the whole syndrome and flag bit histories. If $N_{11} \geq t$, stop the syndrome measurements. Suppose that the latest round is round l. Perform Z-type error correction using $(\vec{s}_{l,x}, \vec{f}_{l-1,z})$, and perform X-type error correction using $(\vec{s}_{l,z}, \vec{f}_{l,x})$.*

The two-tailed adaptive time decoder for flag FTEC developed in this work use similar ideas to the adaptive strong decoder presented in [29]. Therefore, the number of syndrome measurement rounds in the worst-case scenario is $(t+3)^2/4 - 1$ when t is odd, and is $(t+2)(t+4)/4 - 1$ when t is even. This can be proved by assuming that all faults does not cause any nonzero flag bits, then the rest of the proof follows the proof of Theorem 2 in [29].

If the syndrome $\vec{s}_l$ and cumulative flag vector $\vec{f}_l = \sum_{i=1}^{l} \vec{f}_i \pmod 2$ of round l are used for error correction, any faults that happened up to round l will be corrected. However, because round l may correspond to some η_j in the middle of $\vec{\delta}$, an output error may correspond to a nontrivial cumulative flag vector. Therefore, Protocol 2 may not satisfy FTEC conditions in Definition 4 and cannot be applied to fault-tolerant quantum computation. Nevertheless, Protocol 2 is still applicable to a quantum memory. To do so, one needs to pass the *remaining cumulative flag vector* of the current FTEC routine (the sum of the flag vectors from round $l + 1$ onward) to the next FTEC routine and use it as an initial flag vector.

3. One-tailed adaptive time decoder

One-tailed and two-tailed decoders use similar ideas to estimate the number of faults, except that in the one-tailed case the syndrome and cumulative vector for error correction must be from the very last zero substring in $\vec{\delta}$ (it is to ensure that the output error satisfies the both conditions in Definition 4). Suppose that

$\vec{\delta} = \eta_1 1 \eta_2 1 \ldots 1 \eta_c$ for some positive integer c, η_c has length $\gamma_c \geq 1$, and bit one on the left of η_c is the i_1-th bit of $\vec{\delta}$. We define α_c as usual, and define μ_c to be the total numbers of nonzero flag bits obtained from round 1 to round i_1. Also, we define ω_c to be the sum of the numbers of flag bits that exceed 1 bit per round during round $i_1 + 1$ onward. Let $\tilde{\alpha}_c = \max(\alpha_c, \mu_c)$. In this case, the total estimate of the number of occurred faults is $\tilde{\alpha}_c + \omega_c$.

Assuming that there are at most t faults in the whole protocol, if we find that $\tilde{\alpha}_c + \gamma_c + \omega_c \geq t$, at least one round in the $\gamma_c + 1$ rounds that give rise to η_c must have a correct syndrome. This is the first possible stop condition.

The second possible stop condition is similar to what we have for the two-tailed decoder. Let N_{11} be the total number of non-overlapping 11 sequences in the whole $\vec{\delta}$. If $N_{11} \geq t$, the last round must have a correct syndrome.

Suppose that a table-based space decoder for flag FTEC can be constructed, a flag FTEC protocol with the one-tailed adaptive time decoder is as follows:

Protocol 3. *Flag FTEC protocol with one-tailed adaptive time decoder*

Let $t = \lfloor (d-1)/2 \rfloor$ be the number of errors that a stabilizer code of distance d can correct. Let $\vec{s}_i = (\vec{s}_{i,x}, \vec{s}_{i,z})$ and $\vec{f}_i = (\vec{f}_{i,x}, \vec{f}_{i,z})$ be syndrome and cumulative flag vector obtained from the i-th round of full syndrome measurements with flag circuits. After the i-th round with $i \geq 2$, calculate δ_{i-1}. Repeat syndrome measurements until one of the following conditions is satisfied:

1. $\tilde{\alpha}_c, \gamma_c, \omega_c$ satisfy $\tilde{\alpha}_c + \gamma_c + \omega_c \geq t$;

2. $N_{11} \geq t$.

Suppose that the latest round when any condition is satisfied is round l. Perform Z-type error correction using $(\vec{s}_{l,x}, \vec{f}_{l-1,z})$, and perform X-type error correction using $(\vec{s}_{l,z}, \vec{f}_{l,x})$.

The number of rounds of full syndrome measurements in the worst-case scenario for Protocol 3, which is also the minimum number of rounds required to guarantee that error correction can be done, can be found by the following theorem:

Theorem 1. *Suppose that flag circuits being used in Protocol 3 give a distinguishable fault set $\mathcal{F}_t$, where $t = \lfloor (d-1)/2 \rfloor$ and d is the distance of the stabilizer code. Performing $\frac{t(t+3)}{2} + 2$ rounds of full syndrome measurements is sufficient to guarantee that Protocol 3 is strongly t-fault tolerant; i.e., both conditions in Definition 4 are satisfied.*

Proof. Suppose that $\vec{\delta} = \eta_1 1 \eta_2 1 \ldots 1 \eta_c$ and $\gamma_c \geq 1$. We will show that if none of η_1, $\eta_1 1 \eta_2$, $\eta_1 1 \eta_2 1 \eta_3$, ..., $\eta_1 1 \eta_2 1 \ldots 1 \eta_c$ satisfies any condition in Protocol 3, the maximum length of such $\vec{\delta}$ is $\frac{t(t+3)}{2}$. In the worst-case

scenario, flag measurement results do not help in estimating the number of occurred faults, so we can assume that $\tilde{\alpha}_c = \alpha$ and $\mu_{\text{all}-1}$. Below are the results from analyzing η_1, $\eta_1 1 \eta_2$, and $\eta_1 1 \eta_2 1 \eta_3$:

1. For η_1, $\alpha_c = 0$ and $\gamma_c = \gamma_1$, so the maximum length of η_1 such that $\tilde{\alpha}_c + \gamma_c \geq t$ is not satisfied is $t - 1$.

2. For $\eta_1 1 \eta_2$, $\alpha_c = 0$ and $\gamma_c = \gamma_2$, so the maximum length of η_2 such that $\tilde{\alpha}_c + \gamma_c \geq t$ is not satisfied is $t - 1$.

3. For $\eta_1 1 \eta_2 1 \eta_3$, $\alpha_c = 1$ and $\gamma_c = \gamma_3$, so the maximum length of η_3 such that $\tilde{\alpha}_c + \gamma_c \geq t$ is not satisfied is $t - 2$.

By induction, the maximum length of $\vec{\delta} = \eta_1 1 \eta_2 1 \ldots 1 \eta_c$ such that $\tilde{\alpha}_c + \gamma_c \geq t$ is not satisfied is $(t-1) + 1 + (t-1) + 1 + (t-2) + 1 + \cdots + 1 + 1 + 0 + 1$, which is $\frac{t(t+3)}{2}$. Here $\vec{\delta}$ is of the form,

$$\underbrace{00\ldots00}_{t-1}1\underbrace{00\ldots00}_{t-1}1\underbrace{00\ldots00}_{t-2}1\underbrace{00\ldots00}_{t-3}1\ldots1001011$$
(11)

The number of rounds that gives $\vec{\delta}$ of the maximum length is $\frac{t(t+3)}{2} + 1$. By performing one more round of syndrome measurements, $\vec{\delta}$ is extended by one bit, which must be 0 if the total number of faults is no more than t. In that case, $\tilde{\alpha}_c + \gamma_c \geq t$ will be satisfied. Therefore, $\frac{t(t+3)}{2} + 2$ rounds of full syndrome measurements is sufficient to guarantee that flag FTEC can be performed.

Note that there are other forms of $\vec{\delta}$ in which none of η_1, $\eta_1 1 \eta_2$, $\eta_1 1 \eta_2 1 \eta_3$, ..., $\eta_1 1 \eta_2 1 \ldots 1 \eta_c$ satisfies any condition in Protocol 3, and the length of $\vec{\delta}$ is $\frac{t(t+3)}{2} - 1$; For example, suppose that $t = 3$. Possible forms of such $\vec{\delta}$ are 001101011, and 001001111. In any case, one of the conditions in Protocol 3 will be satisfied if one more round of syndrome measurements is done, so the number of rounds to guarantee fault tolerance is still $\frac{t(t+3)}{2} + 2$. $\square$

Note that the number given by Theorem 1 is worse than that of the two-tailed decoder because we are not allowed to check whether the syndrome of any round in the middle can be used for error correction.

An advantage of the FTEC protocol with one-tailed adaptive time decoder is that it is applicable to any kind of fault-tolerant quantum computation as long as the corresponding fault-tolerant implementation satisfies the revised definitions of fault tolerance in [26]. This is possible because when the syndrome and cumulative flag vector for error correction are from the last zero substring in $\vec{\delta}$, it is guaranteed that the output error corresponds to a zero cumulative flag vector.

C. Extended techniques for CSS codes

In this section, we discuss two additional techniques which can further improve our flag FTEC protocols with adaptive time decoding. The first technique is the separated X and Z counting which is applicable to any CSS code. This technique is based on the ideas from [28] and [29], and can be used to improve the pseudothreshold. The second technique is the classical processing of the remaining cumulative flag vector. This technique allows our flag FTEC protocol with the two-tailed adaptive time decoder to be applicable to any fault-tolerant Clifford computation.

1. Separated X and Z counting

For any CSS code, Z-type and X-type errors can be corrected separately. It is possible to improve the number of measurements by separating the X-type and Z-type syndrome measurement rounds (which correspond to X-type and Z-type stabilizer generators). In this section, we introduce the XZ and ZX decoding strategies. In the XZ strategy, first, we execute a time decoder (which can be Shor, one-tailed, or two-tailed decoder) using only the X-type syndromes. The difference vector for this process is denoted by $\vec{\delta}_x$. After the decoder returns the X-type syndrome and the cumulative flag vectors for Z-type error correction, we estimate the number of faults t_x that could cause $\vec{\delta}_x$; we define $\alpha_{\text{all},x}$ to be the total number of non-overlapping 11 sequences plus the total number of remaining 1s in $\vec{\delta}_x$, define $\mu_{\text{all},x}$ to be the total number of nontrivial flag bits in $\vec{\delta}_x$, and let $t_x = \max(\alpha_{\text{all},x}, \mu_{\text{all},x})$. Given that we spend this number of faults from our fault budget t, we can reduce the target number of faults in the stop condition for the Z-type syndrome measurements. Afterward, we run a time decoder for Z-type syndromes with the target number of faults $t_z = t - t_x$. The ZX strategy is similar to the XZ strategy, except that the Z-type generators are measured first.

When the separated X and Z counting technique is applied to a flag FTEC protocol, one can find syndromes for Z-type and X-type error corrections faster compared to a conventional method where the target numbers of faults for both types of error corrections are t. However, a drawback is that the flag FTEC protocol will only be compatible with quantum memory. This is because of each type of error correction requires flag information of the opposite type. In particular, Suppose that the time decoder for X-type syndrome measurements give syndrome $\vec{s}_x$ and cumulative flag vector $\vec{f}_x$, and the time decoder for Z-type syndrome measurements give syndrome $\vec{s}_z$ and cumulative flag vector $\vec{f}_z$. Z-type error correction will be done by applying a space decoder to $\vec{s}_x$ and the zero cumulative flag vector, while X-type error correction will be done by applying a space decoder to $\vec{s}_x$ and $\vec{f}_z$. The cumulative flag vector $\vec{f}_x$ which has not been used will be treated as the remaining cumulative flag vector of the current FTEC routine and used as an initial flag vector for Z-type error correction in the next FTEC routine.

2. Classical processing of the remaining cumulative flag vector

One drawback of a flag FTEC protocol that uses the two-tailed adaptive time decoder or the separated X and Z counting technique is that it is only applicable to a quantum memory, not a general fault-tolerant quantum computation. This is because the output error at the end of each FTEC routine may correspond to a nontrivial cumulative flag vector. To correct such an error, one needs to pass the flag information from each FTEC routine (the remaining cumulative flag vector) to the next FTEC routine. However, if there are some quantum computation between two FTEC routines (as in an extended rectangle in [5]), the error will be transformed and may not be correctable if the corresponding flag information is not processed properly.

Nevertheless, for any self-dual CSS code, a flag FTEC protocol with two-tailed adaptive time decoder or separated X and Z counting (or both) can made applicable to any fault-tolerant Clifford computation. For example, let us consider an application of a logical Hadamard gate $\bar{H}$ between two FTEC routines. Suppose that the first FTEC routine causes an output error $E_x \cdot E_z$ and the remaining cumulative flag vector is $(\vec{f}_x, \vec{f}_z)$. Without a logical Hadamard gate, E_x and E_z can be corrected using $\vec{f}_z$ and $\vec{f}_x$, respectively. A logical Hadamard gate transforms an X-type error to a Z-type error of the same form, and vice versa. Because the X-type and Z-type generators are of the same form, possible fault combinations for both types of errors are also of the same form. To correct the transformed error $\bar{H}(E_x \cdot E_z)\bar{H}^\dagger$ in the second FTEC routine, one needs to swap the X-type and Z-type cumulative flag vector; that is, the initial flag vector for the second FTEC routine must be $(\vec{f}_z, \vec{f}_x)$.

We can apply similar ideas for flag information processing to logical S and logical CNOT gates. The summary of the classical processing operations for logical H, S, and CNOT gates is provided in Table I. Because $\{H, S, \text{CNOT}\}$ generates the Clifford group, a flag FTEC protocol with two-tailed adaptive time decoder or separated X and Z counting is applicable to any fault-tolerant Clifford computation given that the CSS code is self-dual. Note that the magic state distillation and injection [43, 44] use only Clifford operations. Thus, our techniques are also applicable to fault-tolerant universal quantum computation given that high-fidelity magic states are provided.

Remaining cumulative flag vector of the current FTEC routine	Logical Clifford operation	Initial flag vector of the next FTEC routine
$(\vec{\mathbf{f}}_x, \vec{\mathbf{f}}_z)$	$\bar{H}$	$(\vec{\mathbf{f}}_z, \vec{\mathbf{f}}_x)$
$(\vec{\mathbf{f}}_x, \vec{\mathbf{f}}_z)$	$\bar{S}$	$(\vec{\mathbf{f}}_x, \vec{\mathbf{f}}_x \oplus \vec{\mathbf{f}}_z)$
$(\vec{\mathbf{f}}_{x,1}, \vec{\mathbf{f}}_{z,1} \| \vec{\mathbf{f}}_{x,2}, \vec{\mathbf{f}}_{z,2})$	$\overline{\mathrm{CNOT}}_{1,2}$	$(\vec{\mathbf{f}}_{x,1}, \vec{\mathbf{f}}_{z,1} \oplus \vec{\mathbf{f}}_{z,2} \| \vec{\mathbf{f}}_{x,1} \oplus \vec{\mathbf{f}}_{x,2}, \vec{\mathbf{f}}_{z,2})$

TABLE I: A list of required classical processing operations on the remaining cumulative flag vector in case that a logical Clifford gate is performed between two FTEC routines. With these operations, a flag FTEC protocol with two-tailed adaptive time decoder or separated X and Z counting is applicable to any fault-tolerant Clifford computation, given that the CSS code is self-dual.
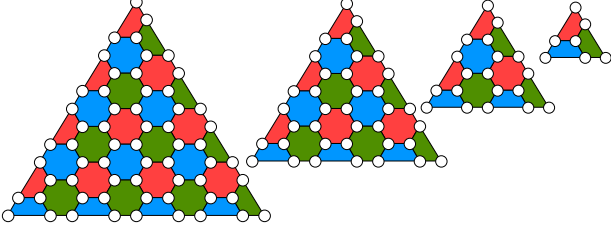
FIGURE 6: The studied members of the hexagonal color code, for distances 3, 5, 7, and 9 (right to left). Qubits are on the vertices and stabilizer generators are the plaquettes. As the codes are self-dual CSS codes, both the X and Z stabilizer generators are described by the same layout.

V. NUMERICAL RESULTS

A. Methods

Our optimization tools for space and time decoders including the compact lookup table construction, the MIM technique, and the adaptive time decoders for flag FTEC are applicable to any stabilizer code. However, we focus on a specific family of codes where the aforementioned tools can be simplified and extended techniques including separated X and Z decoding and classical processing of flag information are applicable—the family of self-dual CSS codes in which the number of physical qubits is odd, the number of logical qubits is 1, and logical X and Z operators are transversal. To evaluate the performance of our tools, we simulate FTEC protocols on the $[[(3d^2 + 1)/4, 1, d]]$ hexagonal color codes [31] of distance 3, 5, 7 and 9. These codes are planar topological codes with configurations displayed in Fig. 6. For each code, stabilizer generators are measured using the syndrome extraction circuits with single flag ancilla, as depicted in Fig. 4. It was proved in [23] that for the hexagonal code of any distance, using flag circuits of this form preserves the code distance regardless of the gate orderings (an alternative proof can be found in [26]). The simulation is implemented under the circuit-level depolarizing noise model specified in Section II A. As there is no idling noise in our error model, the syndromes can be extracted sequentially.

To construct a lookup table for space decoding and to verify that our circuit configurations preserve the code distance, we implement the algorithm described in Section III A using C++. The timing for verification alongside the statistics of the lookup table can be found in Table II. The lookup table for these codes can be generated on the fly before the sampling starts as the required time is low enough.

Here we simulate the storage (i.e. the result of the logical identity operation) of the logical state $|\bar{0}\rangle$. We use of the Pauli frame simulator in Stim [45] to collect measurement samples, and use Cirq [46] for constructing the circuits with the given noise model. After a perfect preparation of $|\bar{0}\rangle$, we perform noisy error correction and recovery. In the error correction process, full rounds of syndrome measurements are repeated until the stop condition of the time decoder is satisfied. The time decoder returns an accepted full syndrome (consisting of error syndrome and cumulative flag vector), then the space decoder determines the recovery operation based on the accepted full syndrome. This recovery operation is applied to the data qubits afterwards. Finally, we apply an ideal error correction and determine whether the output error is a logical X error (which corresponds to having $|\bar{1}\rangle$ as the output state).

B. The overall effect of optimization tools

We first compare two protocols: (1) the FTEC protocol with Shor time decoder without the MIM technique (the protocol in which none of our optimization tools are applied) and (2) the FTEC protocol with the MIM technique and the two-tailed adaptive time decoder with the ZX strategy (the best FTEC protocol in this work which is compatible with any Clifford computation on a self-dual CSS code). The logical error rate p_L vs physical error rate p for hexagonal color codes of distance 3, 5, 7, and 9 are plotted in Fig. 7. Our results show that for each code, applying the optimization tools can significantly improve the pseudothreshold (the intersection between each plot and the $p_L = 2p/3$ line). Furthermore, the optimized decoder yields orders of magnitude improvements in the logical error rate in the $p = 10^{-4}$ error regime.

	$[\![7,1,3]\!]$	$[\![19,1,5]\!]$	$[\![37,1,7]\!]$	$[\![61,1,9]\!]$
# columns of $H_\mathrm{f}|_{H_\mathrm{f}^{-1}}$	28	88	181	307
# unique columns	20	62	128	218
# fault combinations	20	1,953	349,632	93,263,997
Cache size	20	1,587	262,500	67,166,572
Memory	$\leq$ 1kB	$\leq$ 1kB	$\approx$ 50MB	$\approx$ 1.38GB
Verification time	$\leq$ 1ms	$\leq$ 1ms	$\approx$ 720ms	$\approx$ 58.9s

TABLE II: Metrics of the lookup table. The number of columns of the fault check matrix counted in the first row results from the three-part structure of data errors, flag errors, and gate faults. These columns are not necessarily unique, which can be seen in the second row that counts the number of unique columns. The time to verify distinguishability for the different codes on a single thread with our C++ code depends on the number of unique columns, hence the verification of the higher distance code takes longer than shorter ones. All timings are reported using Intel Xeon Gold 6226R, 2.90GHz processors. Some fault combinations have the same full syndrome, hence the cache size is smaller than the full number of fault combinations. The cache size in memory is reported from actual usage, including the overhead of the hash table implementation.

Under a noise model parameterized by a single parameter p, the fault-tolerant threshold p_{th} is the error probability under which the logical error rate is guaranteed to decrease with increasing code distance for a specific code family and decoder. Our decoders can yield a p_{th} for concatenated code families using a level-by-level decoder, but they will not yield a threshold for topological code families for two reasons. The practical reason is that our space decoder that uses a lookup table is not scalable to the large d limit. The fundamental reason is that the time decoder will always take $\vec{\delta}$ in which all bits are one when d is large, because δ_j for each round will be 1 with a probability exponentially close to 1 for finite p. The space decoder then acts on the final state but lacks the information about correlations to properly correct it. This is why an efficient space-time decoder is critical for achieving p_{th} for topological codes.

We can define an effective threshold $\tilde{p}_{th}$ as the error rate below which increasing the code distance improves the logical error rate for this finite set of codes. The optimized protocol yields a $\tilde{p}_{th} = 1.5 \times 10^{-3}$, while the unoptimized protocol yields $\tilde{p}_{th} = 4.5 \times 10^{-5}$. We also note that the crossing point between the codes of distances d and $d-2$ is dropping quickly with the unoptimized decoder, while it is stable for the optimized decoder over this code set. In the next sections, we further discuss the effect of each technique that can contribute to this improvement.

C. The effect of the Meet-in-the-Middle technique

In this section, we evaluate the performance of simulated storage that use the space decoder with and without the MIM technique. We explore the effect for codes of distances 3, 5, 7, and 9, and compare the effect when the time decoder is Shor, one-tail, or two-tail time decoder. We observe a significant decrease in logical error rates and an improvement in pseudothreshold when the

MIM technique is applied. We also find that the benefit increases with the code distance. In Fig. 8, we show the improvement for the code of distance 9 where the benefit is the largest. The results for codes of other distances are provided in Fig. 12.

It is clear that both non-adaptive (Shor) and adaptive time decoders benefit from the MIM technique. The pseudothreshold for the Shor time decoder increases by more than 100%, from $(1.34 \pm 0.01) \times 10^{-4}$ to $(2.79 \pm 0.01) \times 10^{-4}$. The pseudothreshold for one-tailed adaptive time decoder increases by 85%, from $(2.11 \pm 0.05) \times 10^{-4}$ to $(3.91 \pm 0.26) \times 10^{-4}$. Finally, the pseudothreshold for the two-tailed adaptive time decoder gets a boost of 76%, from $(3.38 \pm 0.17) \times 10^{-4}$ to $(5.96 \pm 0.71) \times 10^{-4}$.

D. The effect of the adaptive time decoders

In this section, we compare the performance of the simulated storage numerical experiments that use different time decoders when the MIM technique is applied. The results are displayed in Fig. 9 for the hexagonal code of distance 9, and we refer the reader to Fig. 15 in Appendix A for the results for the codes of other distances.

For the code of distance 9, in comparison with the Shor time decoder, the one-tailed adaptive time decoder improves the pseudothreshold by 40% from $(2.79 \pm 0.07) \times 10^{-4}$ to $(3.91 \pm 0.07) \times 10^{-4}$. The two-tailed method achieves $(5.96 \pm 0.71) \times 10^{-4}$ pseudothreshold, which is more than a 100% increase compared to the Shor time decoder. However, this gain vanishes at lower error rates, and the performances of Shor and one-tailed decoders become similar at around $p = 10^{-4}$. It is not surprising as we expect all adaptive time decoders to converge to Shor time decoder at lower error rates. The main reason for this convergence is that the performance gains for the adaptive techniques come from a decrease in the average number of rounds for syndrome measurements, and the decrease converges to zero at low error rates. How fast the decrease converges does matter, and in contrast
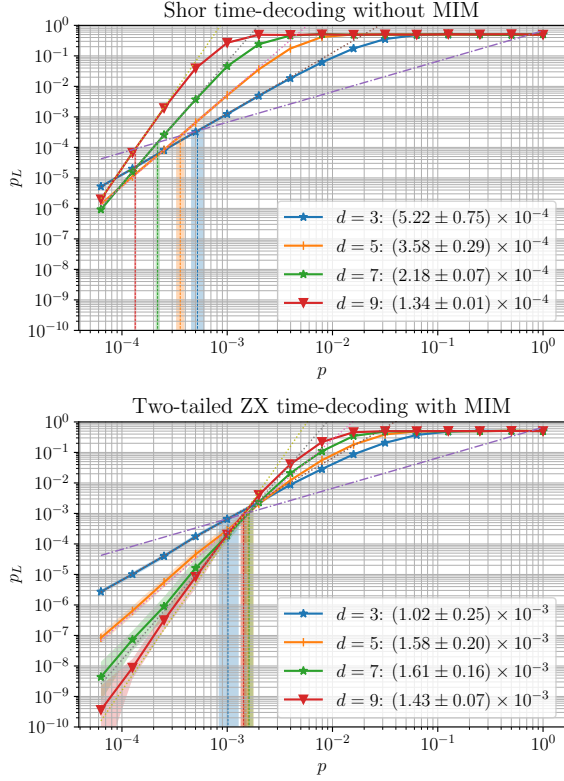
FIGURE 7: The upper plot shows the curve of logical error rate p_L vs physical error rate p for the hexagonal color code family without any of our optimization techniques, using the Shor time decoder without MIM. The lower plot uses the best-performing combination of our techniques, including MIM and the two-tailed adaptive time decoder with ZX strategy. Pseudothresholds for each curve (the intersection points with the $p_L = 2p/3$ line) are included in the labels and marked with vertical lines. The data points represent the number of logical errors divided by the total number of samples at that p error rate and thus estimate the true logical error rates, which should lie within the shaded areas with high confidence. The dotted helper lines marking the expected slope for curves with different distances show good agreement with distance preservation.

to the one-tailed approach, the two-tailed time decoder preserves its performance gain over Shor time decoder at the observed low-error regime as low as 5×10^{-5}.

We also provide the plots of the average numbers of full rounds of measurements for all decoders. At a low-error-rate regime, all decoders have the same minimum number of measurement rounds, $t + 1$, which corresponds to the case that all bits in the difference vector are zeros. We can see the separation more clearly the physical error rate is in the 10^{-3} range; the two-tailed time decoder requires the fewest rounds, followed by the one-tailed decoder, and the Shor time decoder performs the worst. At the high-error-rate regime, all bits in the difference vector tend to be ones. In this case, the Shor time decoder
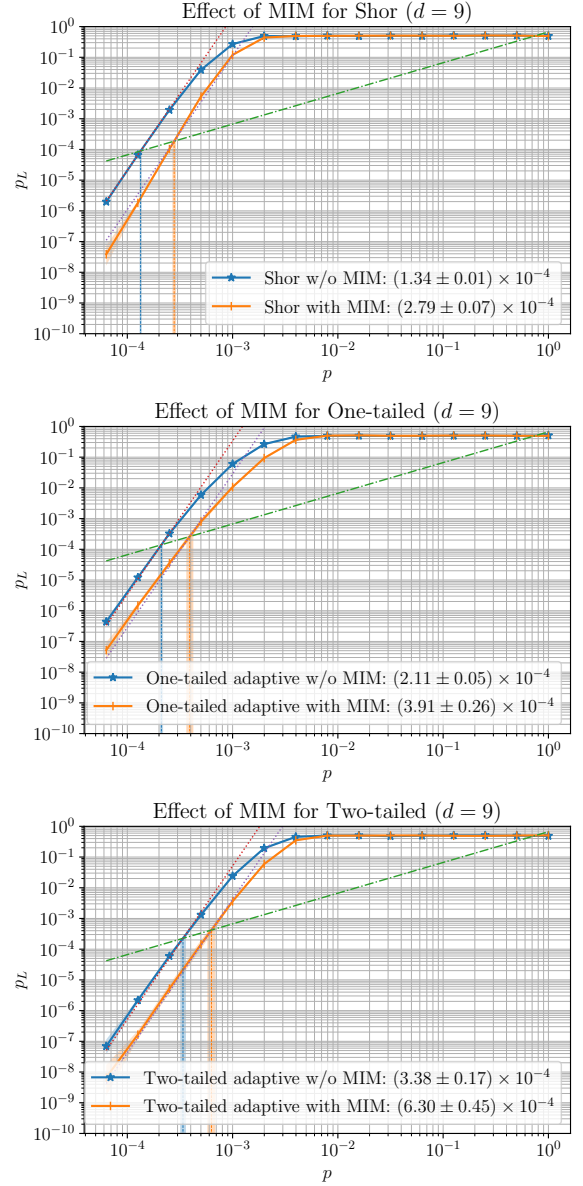
FIGURE 8: The effect of the MIM technique on different time decoders at distance 9. The effect is the largest for Shor time decoder, more than doubling the pseudothreshold. The MIM technique also gives at least a significant 70% improvement on the adaptive time decoders.

requires $(t + 1)^2$ rounds, while both one-tailed and two-tailed decoders requires $2t + 1$ rounds.

E. The effect of the separated X and Z counting technique

In this section, we observe the performance gains when the separated X and Z counting technique is applied. Here we compare the FTEC protocols that use the two-tailed adaptive time decoder with joint X and Z gener-
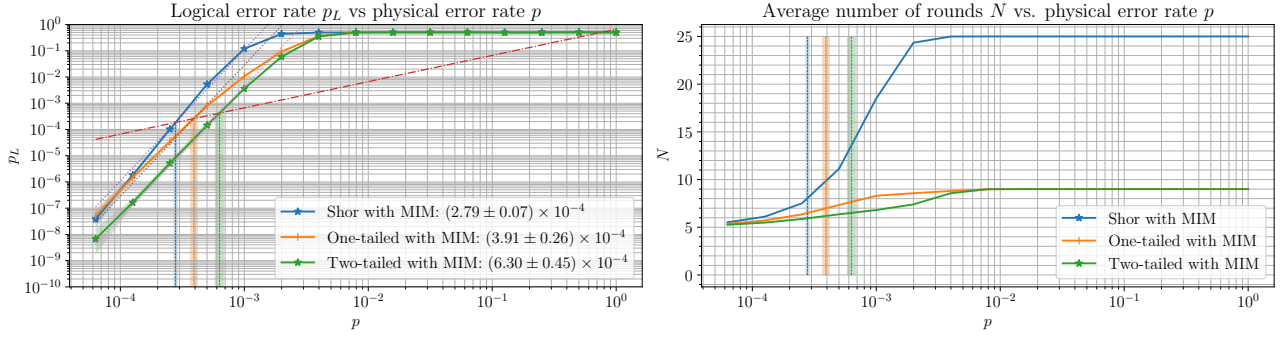
FIGURE 9: Logical error rates of one-tailed and two-tailed adaptive time decoders compared to the Shor time decoder (left) with corresponding average number of rounds (right) for the hexagonal color code of distance 9.
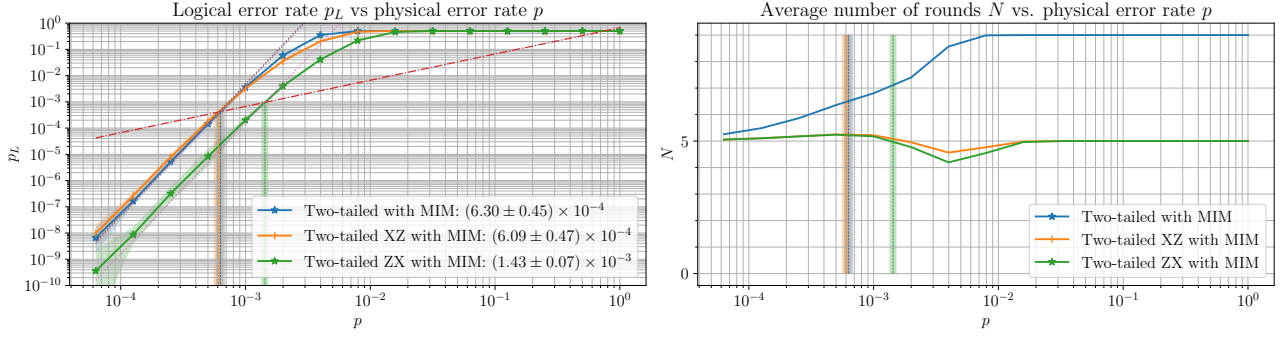

FIGURE 10: Logical error rates of the two-tailed time decoder with XZ and ZX strategies in comparison with the two-tailed adaptive time decoder with joint X and Z measurements (left) and corresponding average number of rounds (right) for the hexagonal color code of distance 9.

ator measurements (as in Section IV B 2), the two-tailed adaptive time decoder with XZ strategy, and the two-tailed adaptive time decoder with ZX strategy (as in Section IV C 1). The logical error rate is calculated from the number of samples in which the output error is a logical X error. The p_L versus p plots for the code of distance 9 are shown in Fig. 10 (the results for codes of other distances can be found in Fig. 16 in Appendix A).

In terms of the pseudothreshold, we observe that the decoder with separated X and Z counting performs the best when Z-type generators are measured before X-type generators. Compared to the two-tailed decoder with joint X and Z generator measurements, the separated two-tailed ZX decoder improves the pseudothreshold by 140% from $(5.96 \pm 0.71) \times 10^{-4}$ to $(1.44 \pm 0.20) \times 10^{-3}$. This is mainly because measuring generators of the first type (X or Z) requires more rounds, and it is more probable that the measurements can cause correlated errors of the same type as the generators being measured (which are more difficult to correct than uncorrelated errors since they require flag information). As in our simulations we measure the performance of storing the logical $|0\rangle$ state (thus, a logical X error is counted), the decoder that measures X-type generators first performs worse. We also observe that there is no significant difference between the two-tailed decoder with joint measurements and the two-tailed decoder with XZ strategy.

We also provide plots of the average number of full rounds of measurements for all decoders (where the full round of single-type generator measurements is counted as $1/2$ full rounds of total measurements). At the low-error-rate regime, all decoders require $t + 1$ rounds. For the original two-tailed decoder, the average number of rounds increases as the physical error rate increases, and it reaches $2t+1$ rounds at the high-error-rate regime. For both two-tailed decoders with separated X and Z counting, we find that the average number of rounds increases near the pseudothreshold, then there are the dips after the pseudothreshold, and the numbers reach $t+1$ rounds at the high-error-rate regime. The dips come from the fact that the measurements of generators of the first type (either X or Z) can stop at less than $(2t + 1)/2$ rounds but the estimate of the number of occurred faults can be t, which then causes the measurements of generators of the second type to stop at $1/2$ rounds. At the high-error-rate regime, the decoders with separated X and Z counting require $t + 1$ rounds since measuring generators of the first type requires $(2t + 1)/2$ rounds while measuring generators of the second type requires $1/2$ rounds on average. Overall, the decoder that measures Z-type generators first performs better than the decoder that measures X-type generators first.

VI. DISCUSSIONS AND CONCLUSIONS

In this work, we focus on flag FTEC with lookup table decoding. We consider an error correction protocol consisting of two parts, the time decoder, and the space decoder. The time decoder finds a correct syndrome from the history of syndromes obtained from repeated syndrome measurements, and the space decoder finds a recovery operator from the correct syndrome. We develop several optimization tools for both space and time decoders which make flag FTEC with lookup table decoding more practical for codes of small distances. Here we focus primarily on the family of self-dual CSS codes in which the number of physical qubits is odd, the number of logical qubits is 1, and logical X and Z operators are transversal, although most of our techniques can be generalized for other stabilizer codes. Here we assume that all stabilizer generators are measured with flag circuits with a single ancilla. We also evaluate the performance of our tools on the hexagonal color codes of distances 3, 5, 7, and 9.

For the space decoder, we first develop a technique to build the lookup table more efficiently in Section III A. With our lookup table construction method, the lookup table for a self-dual CSS code requires 75% less memory compared to the lookup table for a generic stabilizer code. The construction method also verifies the distinguishability of the fault set corresponding to flag circuits for syndrome measurements. This verification can tell whether particular gate orderings in the flag circuits preserve the code distance. Our construction also leads to the notion of the fault code, a linear code corresponding to the faults under circuit-level noise which simplifies the verification of the distance of the protocol. More efficient decoding schemes for the fault code can be an interesting avenue to explore in future work. The fault code can also be useful for improving the performance of numerical simulations of error correction schemes under circuit-level noise.

Another optimization tool for space decoding is the MIM technique in Section III B, which could improve decoding accuracy when the number of faults in the protocol is greater than t (where $t = \lfloor (d-1)/2 \rfloor$ for the code of distance d). The effect of the MIM technique on the simulated storage of the hexagonal color codes is discussed in Section V C (see also Fig. 12). We find that for any kind of time decoder, the logical error rates are reduced and the pseudothresholds are improved when applying the MIM technique. Moreover, the MIM technique becomes more beneficial as the code distance increases.

For the time decoder, we generalize the adaptive syndrome measurement techniques proposed in [29] to flag FTEC, and develop one-tailed and two-tailed adaptive time decoders in Section IV B. For a general stabilizer code in which flag FTEC is possible, the one-tailed decoder is preferable as it is compatible with any fault-tolerant quantum computation, while the two-tailed decoder is applicable to quantum memory only. Neverthe-

less, for self-dual CSS codes, the two-tailed decoder is applicable to any fault-tolerant Clifford computation (and any quantum computation if high-fidelity magic states are provided) with the help of the classical processing technique on cumulative flag vectors developed in Section IV C. The effect of the adaptive time decoders on the simulated storage is discussed in Section V D. We observe that our adaptive time decoders can improve the pseudothresholds compared to the non-adaptive (Shor) time decoder while preserving the code distance. The two-tailed decoder also outperforms the one-tailed decoder.

The two-tailed adaptive decoder without MIM in this work is similar to the adaptive strong decoder in [29], except that this work uses flag circuits instead of syndrome extraction circuits with cat states. The numerical results show that using flag circuits results in a 20-35% increase of the pseudothreshold for the hexagonal color codes of distances 3, 5, 7 and 9. This is mainly because flag circuits have fewer state preparation and qubit measurement locations, although they have more gates. [29] also assumes fault-tolerant preparation of cat states, which requires verification [1] or ancilla decoding circuit [17] that can result in higher space and time overhead. Thus, the pseudothresholds could be worse in that case if additional requirements are also considered. It should be noted that flag circuits may not outperform syndrome extraction circuits with cat states in general, as flag FTEC for other codes may require more complicated flag circuits.

The performance of adaptive time decoders on self-dual CSS codes can be further improved by the separated X and Z counting technique described in Section IV C. Here we estimate the number of occur faults from the measurement of generators of the first type (either X or Z) and then use that information in the measurement of generators of the second type. The effect of this technique can be found in Section V E. When the logical $|0\rangle$ state is stored, we find that the protocol that measures Z-type generators before X-type generators performs the best. We see no significant difference in the protocol that measures X-type generators before Z-type generators, and the protocol that measures X-type and Z-type generators jointly. Thus, the separated X and Z counting provides an advantage only for certain input states depending on the measurement order.

Combining all techniques together, we find a significant improvement in the pseudothreshold while the code distance is still preserved. For example, on the hexagonal color code of distance 9, the pseudothreshold goes up from $(1.34 \pm 0.01) \times 10^{-4}$ to $(1.42 \pm 0.12) \times 10^{-3}$. We also find that in comparison with the unoptimized decoder, the crossing points between the codes of distances d and $d - 2$ come much closer when all techniques are applied (as shown in Fig. 7), leading to a higher effective threshold $\tilde{p}_{th}$ for this set of codes.

While our techniques are applicable to a broader family of codes, it would be interesting to see how our results compare with other works that study error decoding on

the hexagonal color codes under circuit-level noise. For example, Baireuther et al. [47] reported a pseudothreshold above 2×10^{-3} (against $p_L = p$ instead of $p_L = 2p/3$) with a neural-network decoder, which also preserves the code distance empirically. However, it was also reported that training decoders for $d > 7$ became too expensive. By adapting efficient color-decoding algorithms known as restriction decoder [48] and projection decoder [38], Chamberland et al. [23] and Beverland et al. [49] reported threshold values of 2×10^{-3} and 3.7×10^{-3} respectively. The difference between threshold values is mostly contributed by different choices of syndrome extraction circuits: for each weight-six stabilizer generator, Ref. [23] used three flag qubits for connectivity considerations, while Ref. [49] did not use any flag qubit. However, both restriction decoder and projection decoder can only correct up to $d/3$ errors on the color code family considered in this paper[1].

In contrast to the constructions in [23] and [49], our adaptive decoding method preserves the code distance (although the lookup table is not scalable to codes with larger distances). It is expected that our method could become advantageous for the codes of interest when the physical error rate is below a certain value. However, the noise models in Refs. [23, 47, 49] also consider idling noise, while our noise model does not. A fair comparison between the different decoders requires using the same noise model, which is left for future work.

It should be noted that this work uses the adaptive syndrome measurement technique which assumes fast qubit preparation and measurement. For the architectures on which qubit measurement and reset are slow, however, our method may require a large number of ancillas or may not be possible. In that case, one may consider using the flag schemes that do not require fast qubit measurement and reset, such as the flag scheme for any distance-3 code in [50], or the flag scheme in which the flag gadgets are constructed from the classical BCH codes in [51].

VII.   ACKNOWLEDGEMENTS

---

[1] P. W. Shor, "Fault-tolerant quantum computation," *Proceedings., 37th Annual Symposium on Foundations of Computer Science*, pp. 56–65, 1996.

[2] E. Knill, R. Laflamme, and W. H. Zurek, "Resilient quantum computation: error models and thresholds," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1969, pp. 365–384, 1998.

[3] J. Preskill, "Reliable quantum computers," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1969, pp. 385–410, 1998.

[4] D. Aharonov and M. Ben-Or, "Fault-tolerant quantum computation with constant error rate," *SIAM Journal on Computing*, 2008.

[5] P. Aliferis, D. Gottesman, and J. Preskill, "Quantum accuracy threshold for concatenated distance-3 codes," *Quantum Information and Computation*, vol. 6, no. 2, pp. 97–165, 2006.

[6] A. M. Steane, "Overhead and noise threshold of fault-tolerant quantum error correction," *Physical Review A*, vol. 68, no. 4, p. 042322, 2003.

[7] A. Paetznick and B. W. Reichardt, "Fault-tolerant ancilla preparation and noise threshold lower bounds for the 23-qubit Golay code," *Quantum Information and Computation*, vol. 12, pp. 1034–1080, Nov. 2012.

[8] C. Chamberland, T. Jochym-O'Connor, and R. Laflamme, "Overhead analysis of universal concatenated quantum codes," *Physical Review A*, vol. 95, no. 2, p. 022313, 2017.

[9] R. Takagi, T. J. Yoder, and I. L. Chuang, "Error rates and resource overheads of encoded three-qubit gates," *Physical Review A*, vol. 96, no. 4, p. 042302, 2017.

[10] C. Gidney and M. Ekerå, "How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits," *Quantum*, vol. 5, p. 433, 2021.

[11] D. Gottesman, *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology, 1997.

[12] A. Steane, "Multiple-particle interference and quantum error correction," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 452, no. 1954, pp. 2551–2577, 1996.

[13] E. Knill, "Scalable quantum computing in the presence of large detected-error rates," *Physical Review A*, vol. 71, no. 4, p. 042322, 2005.

[14] Y. Tomita and K. M. Svore, "Low-distance surface codes under realistic quantum noise," *Physical Review A*, vol. 90, no. 6, p. 062320, 2014.

[15] S. Huang and K. R. Brown, "Fault-tolerant compass codes," *Physical Review A*, vol. 101, no. 4, p. 042312, 2020.

[16] M. Li, M. Gutiérrez, S. E. David, A. Hernandez, and K. R. Brown, "Fault tolerance with bare ancillary qubits for a [[7, 1, 3]] code," *Physical Review A*, vol. 96, no. 3, p. 032341, 2017.

---

[1] See Appendix A of the 3rd arXiv version of [23].

[17] D. P. DiVincenzo and P. Aliferis, "Effective fault-tolerant quantum computation with slow measurements," *Physical review letters*, vol. 98, no. 2, p. 020501, 2007.

[18] T. J. Yoder and I. H. Kim, "The surface code with a twist," *Quantum*, vol. 1, p. 2, Apr. 2017.

[19] R. Chao and B. W. Reichardt, "Quantum error correction with only two extra qubits," *Physical review letters*, vol. 121, no. 5, p. 050502, 2018.

[20] R. Chao and B. W. Reichardt, "Flag fault-tolerant error correction for any stabilizer code," *PRX Quantum*, vol. 1, no. 1, p. 010302, 2020.

[21] C. Chamberland and M. E. Beverland, "Flag fault-tolerant error correction with arbitrary distance codes," *Quantum*, vol. 2, p. 53, Feb. 2018.

[22] T. Tansuwannont, C. Chamberland, and D. Leung, "Flag fault-tolerant error correction, measurement, and quantum computation for cyclic Calderbank-Shor-Steane codes," *Physical Review A*, vol. 101, no. 1, p. 012342, 2020.

[23] C. Chamberland, A. Kubica, T. J. Yoder, and G. Zhu, "Triangular color codes on trivalent graphs with flag qubits," *New Journal of Physics*, vol. 22, no. 2, p. 023019, 2020.

[24] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, "Topological and subsystem codes on low-degree graphs with flag qubits," *Physical Review X*, vol. 10, no. 1, p. 011022, 2020.

[25] T. Tansuwannont and D. Leung, "Fault-tolerant quantum error correction using error weight parities," *Physical Review A*, vol. 104, no. 4, p. 042410, 2021.

[26] T. Tansuwannont and D. Leung, "Achieving fault tolerance on capped color codes with few ancillas," *PRX Quantum*, vol. 3, no. 3, p. 030322, 2022.

[27] C. Zalka, "Threshold estimate for fault tolerant quantum computation," *arXiv preprint quant-ph/9612028*, 1996.

[28] N. Delfosse and B. W. Reichardt, "Short shor-style syndrome sequences," *arXiv preprint arXiv:2008.05051*, 2020.

[29] T. Tansuwannont, B. Pato, and K. R. Brown, "Adaptive syndrome measurements for Shor-style error correction," *Quantum*, vol. 7, p. 1075, Aug. 2023.

[30] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, "Topological quantum memory," *Journal of Mathematical Physics*, vol. 43, no. 9, pp. 4452–4505, 2002.

[31] H. Bombin and M. A. Martin-Delgado, "Topological quantum distillation," *Physical review letters*, vol. 97, no. 18, p. 180501, 2006.

[32] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Physical Review A*, vol. 54, no. 2, p. 1098, 1996.

[33] D. Poulin, "Optimal and efficient decoding of concatenated quantum block codes," *Physical Review A*, vol. 74, p. 052333, Nov 2006.

[34] S. Huang and K. R. Brown, "Between shor and steane: A unifying construction for measuring error syndromes," *Physical review letters*, vol. 127, no. 9, p. 090505, 2021.

[35] S. Huang and K. R. Brown, "Constructions for measuring error syndromes in calderbank-shor-steane codes between shor and steane methods," *Phys. Rev. A*, vol. 104, p. 022429, Aug 2021.

[36] L. Berent, L. Burgholzer, P.-J. H. S. Derks, J. Eisert, and R. Wille, "Decoding quantum color codes with maxsat," Mar 2023. arXiv:2303.14237 [quant-ph].

[37] N. Maskara, A. Kubica, and T. Jochym-O'Connor, "Advantages of versatile neural-network decoding for topological codes," *Physical Review A*, vol. 99, p. 052351, May 2019.

[38] N. Delfosse, "Decoding color codes by projection onto surface codes," *Physical Review A*, vol. 89, p. 012317, Jan 2014.

[39] O. Higgott and C. Gidney, "Sparse blossom: correcting a million errors per core second with minimum-weight matching," 2023.

[40] A. M. Tillmann and M. E. Pfetsch, "The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing," *IEEE Transactions on Information Theory*, vol. 60, no. 2, pp. 1248–1259, 2013.

[41] D. Tucket, "qecsim is a python 3 package for simulating quantum error correction using stabilizer codes," Aug 2021. https://github.com/qecsim/qecsim.

[42] R. E. Korf, "Search techniques," p. 31–43, 2003.

[43] S. Bravyi and A. Kitaev, "Universal quantum computation with ideal Clifford gates and noisy ancillas," *Physical Review A*, vol. 71, no. 2, p. 022316, 2005.

[44] S. Bravyi and J. Haah, "Magic-state distillation with low overhead," *Physical Review A*, vol. 86, no. 5, p. 052329, 2012.

[45] C. Gidney, "Stim: a fast stabilizer circuit simulator," *Quantum*, vol. 5, p. 497, July 2021.

[46] C. Developers, "Cirq," Dec. 2022. See full list of authors on Github: https://github.com/quantumlib/Cirq/graphs/contributors.

[47] P. Baireuther, M. D. Caio, B. Criger, C. W. J. Beenakker, and T. E. O'Brien, "Neural network decoder for topological color codes with circuit level noise," *New Journal of Physics*, vol. 21, p. 013003, jan 2019.

[48] A. Kubica and N. Delfosse, "Efficient color code decoders in $d \geq 2$ dimensions from toric code decoders," *arXiv preprint arXiv:1905.07393*, 2019.

[49] M. E. Beverland, A. Kubica, and K. M. Svore, "Cost of universality: A comparative study of the overhead of state distillation and code switching with color codes," *PRX Quantum*, vol. 2, no. 2, p. 020341, 2021.

[50] P. Prabhu and B. W. Reichardt, "Fault-tolerant syndrome extraction and cat state preparation with fewer qubits," 2021.

[51] B. Anker and M. Marvian, "Flag gadgets based on classical codes," 2022.

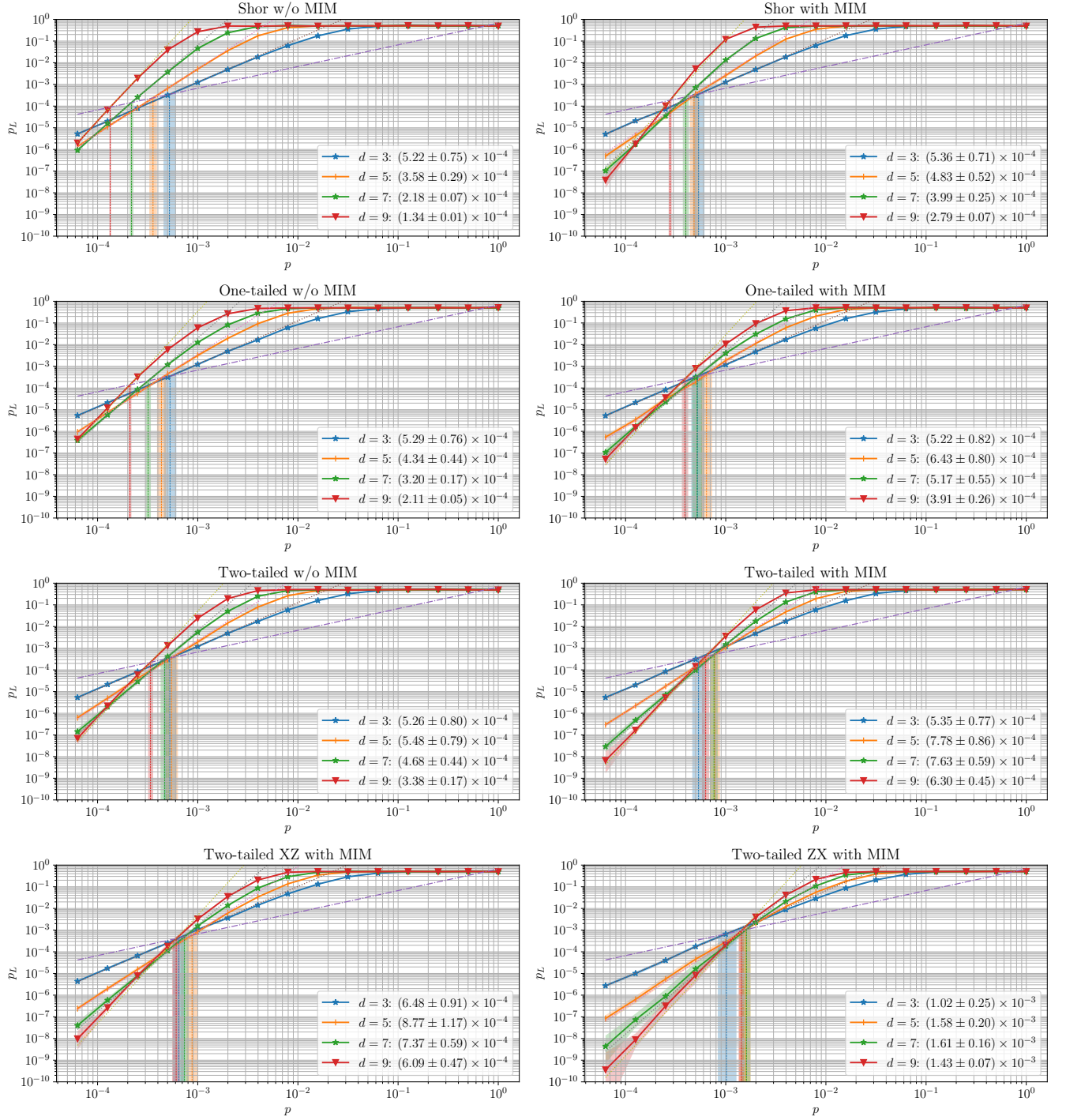**Appendix A: Figures for all distances**

FIGURE 11: The threshold formation effect of increasingly better space and time decoders. Both space decoding improvement (MIM) and time decoding improvements (from Shor to two-tailed ZX-strategy) help in making the intersections of the p_L vs p curves more focused.
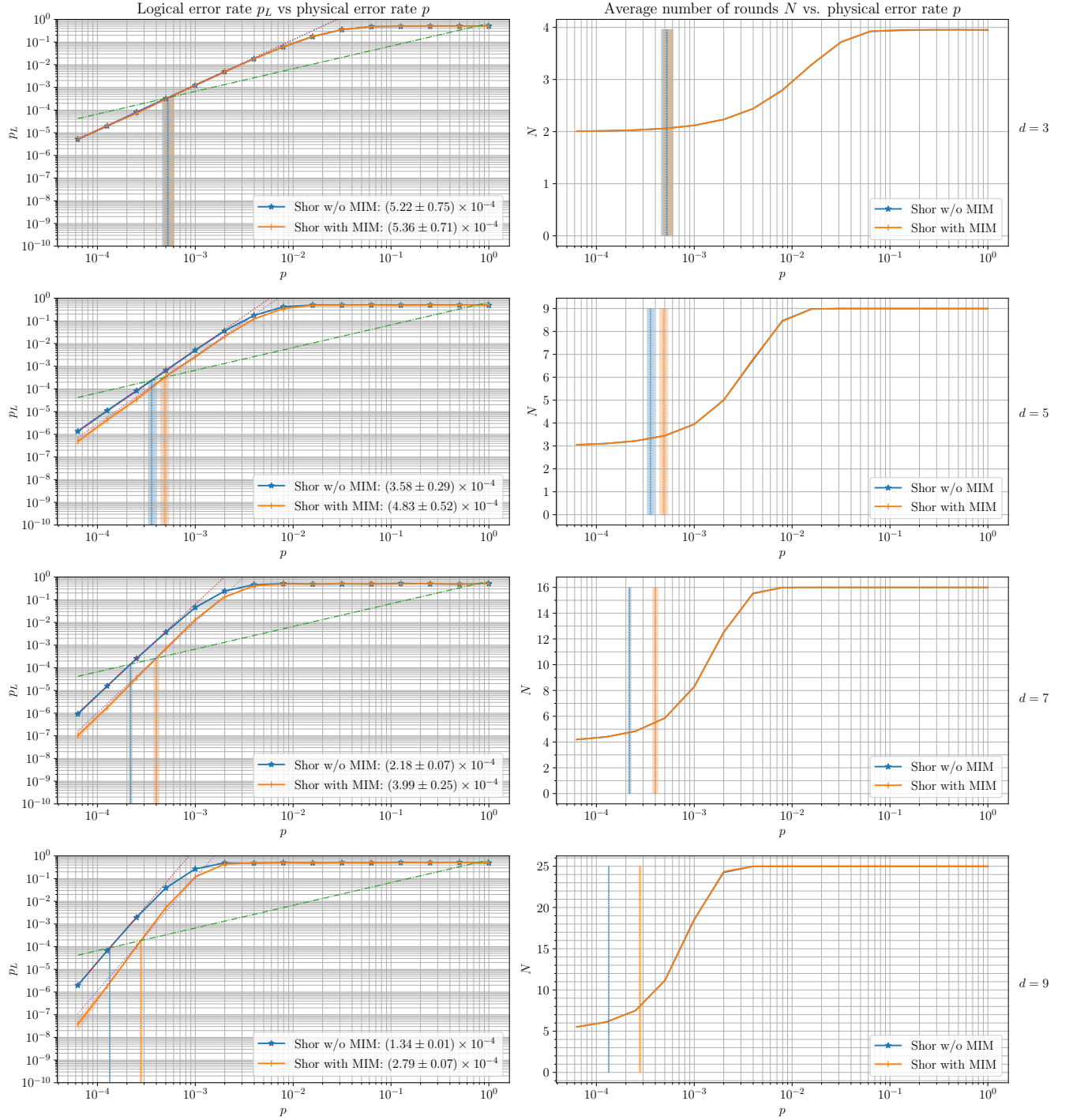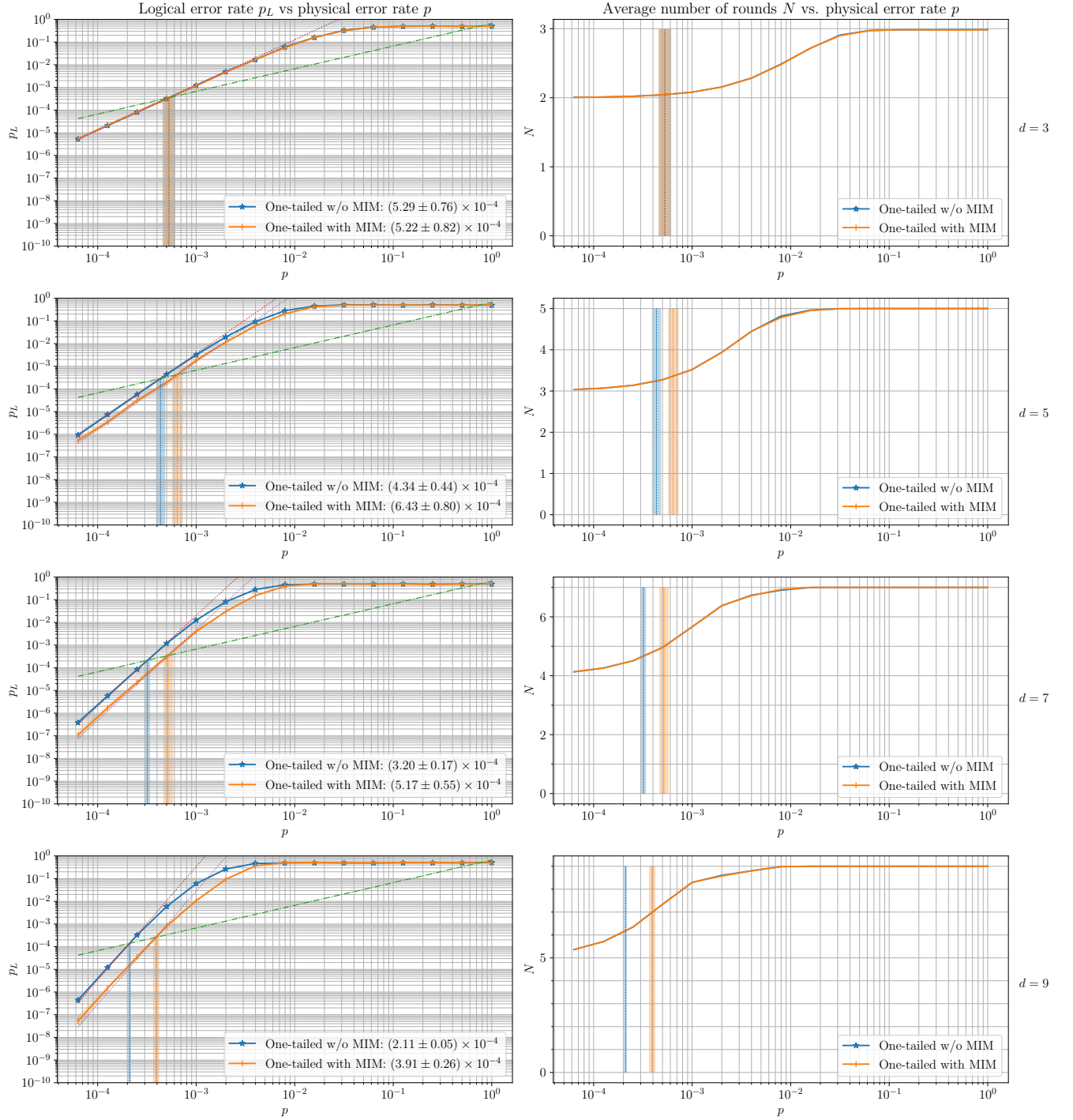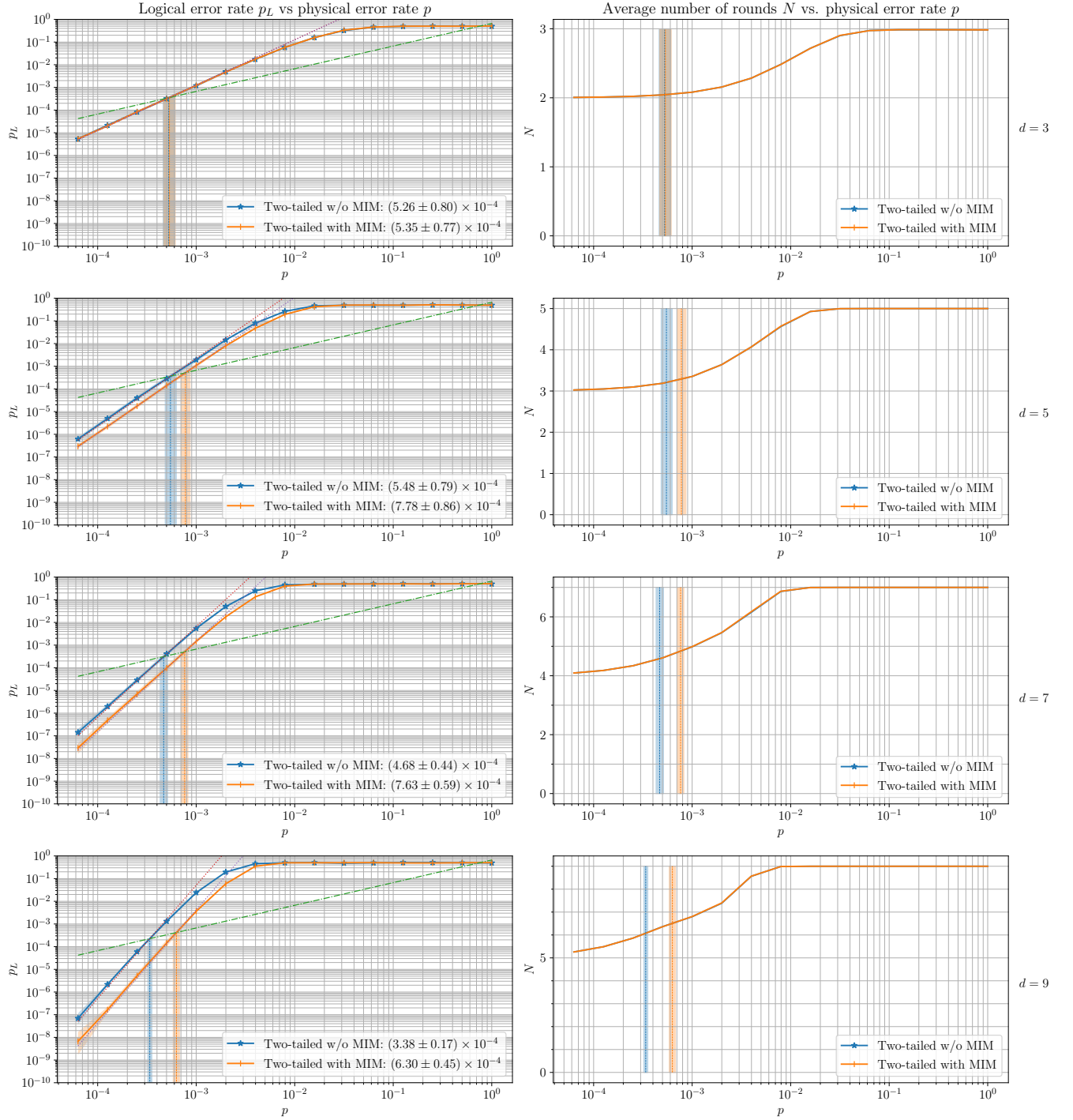
FIGURE 12: The effect of the MIM tecnique on Shor time decoder for hexagonal color codes of distances 3, 5, 7, and 9. The improvement is increasing with the code distance, with no improvement at $d = 3$ and the biggest one at $d = 9$.

FIGURE 13: The effect of the MIM technique on one-tailed adaptive time decoder for hexagonal color codes of distances 3, 5, 7, and 9. The improvement is increasing with distance, with no improvement at $d = 3$ and the biggest one at $d = 9$.

FIGURE 14: The effect of the MIM technique on two-tailed decoder for hexagonal color codes of distances 3, 5, 7, and 9. The improvement is increasing with distance, with no improvement at $d = 3$ and the biggest one at $d = 9$.
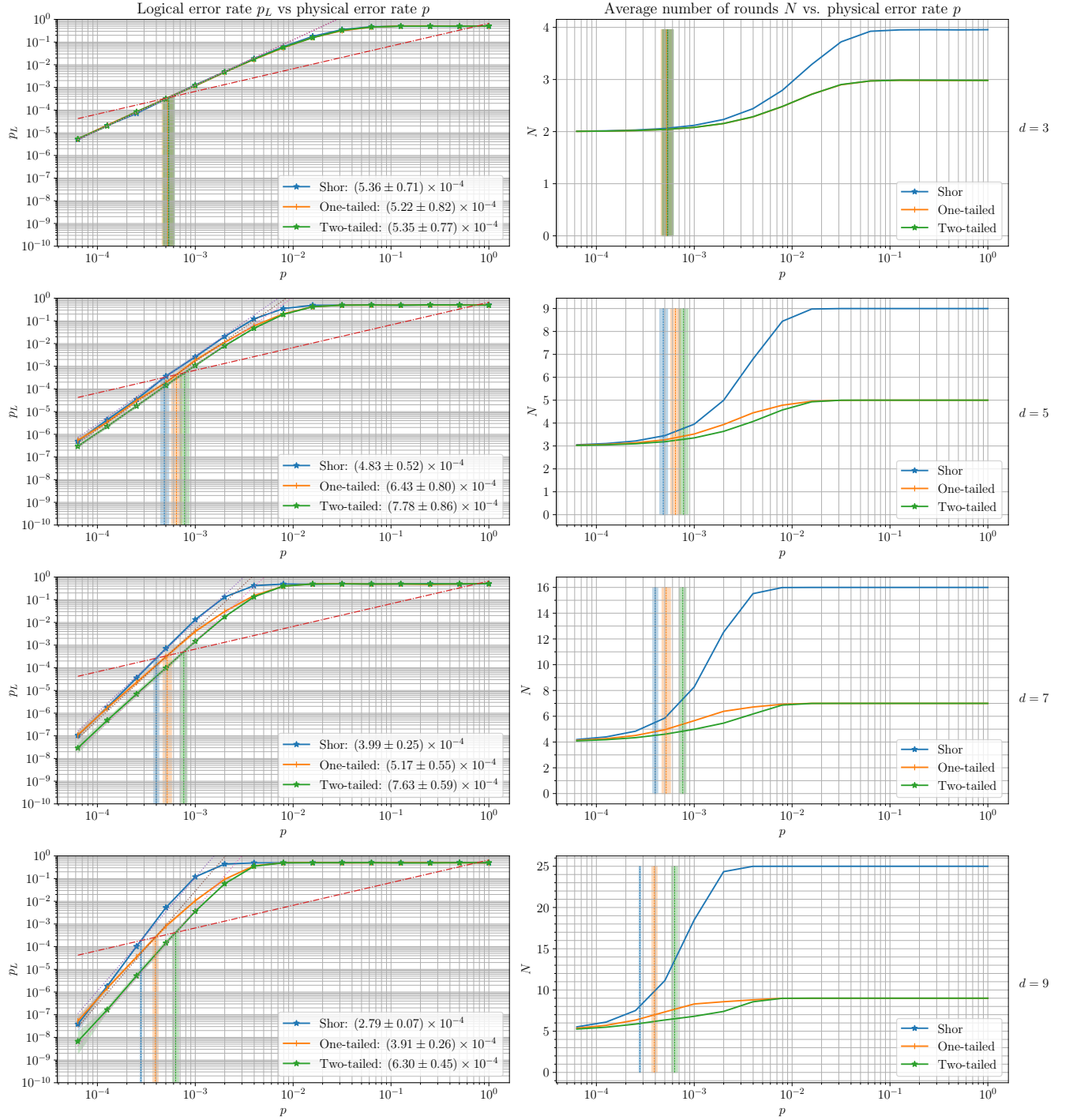
FIGURE 15: Comparison of one-tailed and two-tailed adaptive time decoders to Shor time decoder for hexagonal color codes of distances 3, 5, 7, and 9. The improvement is increasing with distance, with no improvement at $d = 3$ and the biggest one at $d = 9$. Here all decoders use MIM-enhanced space decoding.
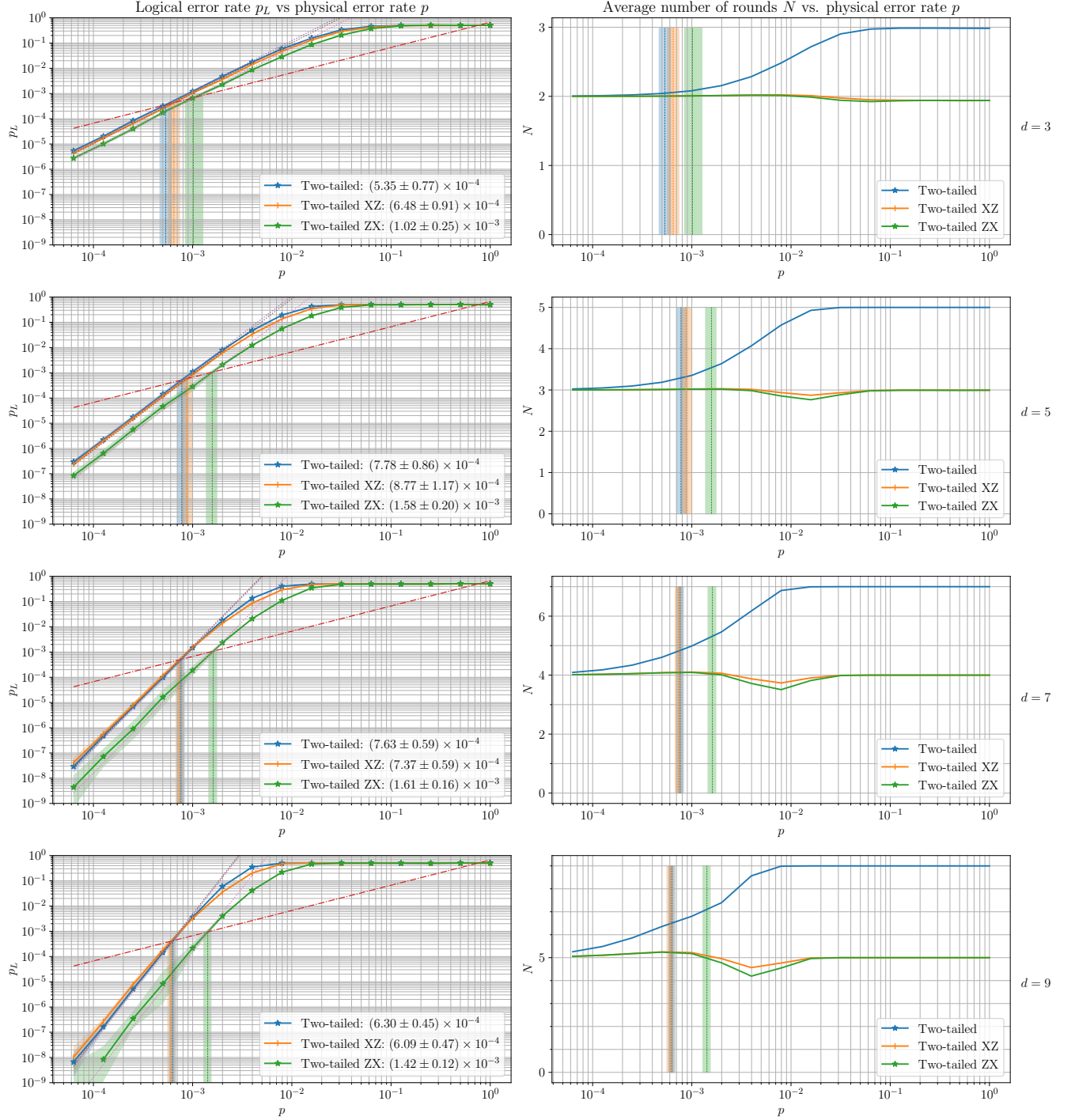
FIGURE 16: Comparison of the two-tailed time decoder with joint measurements and two-tailed time decoders with XZ and ZX strategies for hexagonal color codes of distances 3, 5, 7, and 9. Here all decoders use space decoding with MIM.