Numerical Techniques for Finding the Distances of Quantum Codes

Ilya Dumer*, Alexey A. Kovalev†, and Leonid P. Pryadko‡

* Department of Electrical Engineering, University of California, Riverside, USA (e-mail: dumer@ee.ucr.edu)
† Department of Physics, University of Nebraska at Lincoln, USA (e-mail: alexey.kovalev@unl.edu)
‡ Department of Physics & Astronomy, University of California, Riverside, USA (e-mail: leonid@landau.ucr.edu)

Abstract—We survey the existing techniques for calculating code distances of classical codes and apply these techniques to generic quantum codes. For classical and quantum LDPC codes, we also present a new linked-cluster technique. It reduces complexity exponent of all existing deterministic techniques designed for codes with small relative distances (which include all known families of quantum LDPC codes), and also surpasses the probabilistic technique for sufficiently high code rates.

I. INTRODUCTION

Quantum error correction (QEC) [1]–[3] is a critical part of quantum computing due to fragility of quantum states. To date, surface (toric) quantum codes [4], [5] and related topological color codes [6]–[8] have emerged as prime contenders [9], [10] of efficient quantum design due to two important advantages. Firstly, they only require simple local gates for quantum syndrome measurements, and secondly, they efficiently correct errors below a threshold of about 1% per gate. Unfortunately, the locality also limits [11] such codes to an asymptotically zero rate $k/n$. This would make a useful quantum computer prohibitively large. Therefore, there is much interest in designing of feasible quantum codes with no locality restrictions.

A more general class of codes is quantum low-density-parity-check (LDPC) codes [12], [13]. These codes assume no locality but only require that stabilizer generators (parity checks) have low weight. Unlike surface or color codes, quantum LDPC codes can have a nonzero error probability threshold, both quantum LDPC codes can have a finite rate $k/n$, and also surpasses the probabilistic technique for high-rate codes with small relative distance.

II. BACKGROUND

Let $C[n,k]_{q}$ be a linear $q$-ary code of length $n$ and dimension $k$ in the vector space $\mathbb{F}_q^n$ over the field $\mathbb{F}_q$. This code is uniquely specified by the parity check matrix $H$, namely $C = \{c \in \mathbb{F}_q^n | Hc = 0\}$. Let $d$ denote the Hamming distance of code $C$.

A quantum $[[n,k]]$ (qubit) stabilizer code $Q$ is a $2^k$-dimensional subspace of the $n$-qubit Hilbert space $\mathbb{C}^{2^n}$, a common $+1$ eigenspace of all operators in an Abelian stabilizer group $\mathcal{S}$, where the $n$-qubit Pauli group $\mathcal{P}_n$ is generated by tensor products of the $X$ and $Z$ single-qubit Pauli operators. The stabilizer is typically specified in terms of its generators, $\mathcal{S} = \{S_1, \ldots, S_{n-k}\}$; measuring the generators $S_i$ produces the syndrome vector. The weight of a Pauli operator is the number of qubits it affects. The distance $d$ of a quantum code is the minimum weight of an operator $U$ which commutes with all operators from the stabilizer $\mathcal{S}$, but is not a part of the stabilizer, $U \notin \mathcal{S}$.

A Pauli operator $U \equiv i^n X^v Z^u$, where $v, u \in \{0, 1\}^n$ and $X^v = X_1^{v_1} X_2^{v_2} \cdots X_n^{v_n}$, $Z^u = Z_1^{u_1} Z_2^{u_2} \cdots Z_n^{u_n}$, can be mapped, up to a phase, to a quaternary vector, $e \equiv u + \omega v$, where $\omega^2 \equiv \bar{\omega} \equiv \omega + 1$. A product of two quantum operators corresponds to a sum (mod 2) of the corresponding vectors. Two Pauli operators commute if and only if the trace inner product $e_1 \cdot e_2 \equiv e_1 \cdot \overline{e_2} + \overline{e_1} \cdot e_2$ of the corresponding vectors is zero, where $\overline{e} \equiv u + \overline{\omega} v$. With this map, generators of a stabilizer group are mapped to the rows of a parity check

Note that most error patterns for such codes form small clusters that affect disjoint sets of stabilizer generators [14]. While some errors can have huge weight, they can be always detected if the size of each cluster is below the code distance $d$. We then design an algorithm that verifies code distance by checking the error patterns that correspond to the connected error clusters. For any error weight $w \ll n$, such clusters form an exponentially small fraction of generic errors of the same weight. Therefore, we consider the worst-case scenario that holds for any LDPC code and can be applied in quantum setting. This cluster-based algorithm exponentially reduces the complexity of the known deterministic techniques for sufficiently small relative distance, which is the case for all known families of weight-limited quantum LDPC codes. The new algorithm also outperforms probabilistic techniques for high-rate codes with small relative distance.
matrix $H$ of an additive code over $\mathbb{F}_4$, with the condition that any two rows yield a nil trace inner product $[29]$. The vectors generated by rows of $H$ correspond to stabilizer generators that act trivially on the code; these vectors form the degeneracy group and are omitted from the distance calculation.

An LDPC code, quantum or classical, is a code with a sparse parity check matrix. For a regular $(j, \ell)$ LDPC code, every column and every row of $H$ have weights $j$ and $\ell$ respectively, while for a $(j, \ell)$-limited LDPC code these weights are limited from above by $j$ and $\ell$. A huge advantage of classical LDPC codes is that they can be decoded in linear time using belief propagation (BP) and the related iterative methods $[21], [22]$. Unfortunately, this is not necessarily the case for quantum LDPC codes, which have many short loops of length 4 in their Tanner graphs. In turn, these loops cause a drastic deterioration in the convergence of the BP algorithm $[23]$.

This problem can be circumvented with specially designed quantum codes $[18], [24]$, but a general solution is not known. One alternative that has polynomial complexity in $n$ and approaches linear complexity for very low error rates is the cluster-based decoding of $[14]$.

### III. Generic Techniques for Distance Calculation

The problem of verifying the distance of a linear code (finding a minimum-weight codeword) is related to the decoding problem: find an error of minimum weight that gives the same syndrome as the received codeword. The number of required operations $N$ usually scales as an exponent $N \propto q^{Fn}$ in blocklength $n$, and we characterize the complexity by the exponent $F = \lim_{n \to \infty} (\log_q N)/n$. For example, for a linear $q$-ary code with $k$ information qubits, inspection of all $q^k$ distinct codewords has (time) complexity exponent $F = R$, where $R = k/n$ is the code rate. Given substantially large memory, one can instead consider the syndrome table that stores the list of all $q^{n-k}$ syndromes and coset leaders. This setting gives (space) complexity $F = 1 - R$.

#### A. Sliding window (SW) technique

This technique has been proposed in Ref. $[25]$ for correction of binary errors and generalized in Ref. $[26]$ for soft-decision decoding (where more reliable positions have higher error costs). A related technique has also been considered in Refs. $[27], [28]$. The following proposition addresses this technique for quantum codes. Let $H_q(x) = x \log_q(q - 1) - x \log_q x + (1 - x) \log_q(1 - x)$ be the $q$-ary entropy function. Below we consider both generic stabilizer codes and those that meet the quantum GV bound

$$R = 1 - 2H_4(\delta)$$  \hspace{1cm} (1)

**Proposition 1.** Code distance $\delta n$ of a random quantum stabilizer code $[[n, Rn]]$ can be found with complexity exponent

$$F_{Bq} = (1 + R)H_4(\delta)$$  \hspace{1cm} (2)

For random stabilizer codes that meet the GV bound $[7]$, $F_{Aq} = (1 - R^2)/2$  \hspace{1cm} (3)

**Proof:** SW technique uses only $k + o(n)$ consecutive positions to recover a codeword of a $q$-ary linear $[n, k]$ code. For example, any $k$ consecutive positions suffice in a cyclic code. It is also easy to verify that in most (random) $k \times n$ generator matrices $G$ any $s = k + 2 \log_q n$ consecutive columns form a submatrix $G_s$ of a maximum rank $k$. Thus, in most random $[n, k]$ codes, a codeword can be recovered by encoding its $s$ (error free) consecutive bits. To find a codeword $c$ of any given weight $w$, we choose a sliding window $I(i, s)$ of length $s$ that begins in position $i = 0, \ldots, n - 1$. Note that a sliding window can change its weight only by one when it moves from any position $i$ to $i + 1$; thus at least one of the $n$ windows will have the average Hamming weight $v \equiv \lfloor ds/n \rfloor$. Our algorithm takes all possible positions $i$ and weights $w = 1, 2, \ldots$. We assume that the current window $I(i, s)$ is corrupted in $v$ positions and encode all

$$L = (q - 1)^v \binom{n}{v}$$  \hspace{1cm} (4)

vectors of length $s$ and weight $v$. Procedure stops for some $w$ once we find an encoded codeword $c$ of weight $w$. Finally, such vector $c$ is tested on linear dependence with the rows of the parity check matrix $H$. This gives the overall SW-complexity of order $Ln^2$ with complexity exponent $F_A = RH_4(\delta)$.

To apply SW procedure to a (degenerate) quantum code, note that an $[[n, k]]$ stabilizer code is related to some additive quaternary code that is defined in a space of $4^n$ vectors and has only $2^{n-k} = 4^{r/2}$ distinct syndromes, where $r = n - k$ is the redundancy of the quantum code. Thus, the effective rate is $R' = (n - r/2)/n = (1 + R)/2$, which gives binary complexity exponent $[7]$. Finally, estimate (3) follows from (1).

#### B. Random window (RW) technique $[30]-[32]$

**Proposition 2.** Code distance $\delta n$ of a random quantum stabilizer code $[[n, Rn]]$ can be found with complexity exponent

$$F_{Bq} = H_2(\delta) - \left(\frac{1 - R}{2}\right)H_2\left(\frac{2\delta}{1 - R}\right)$$  \hspace{1cm} (5)

**Proof:** Given a random $q$-ary linear $[n, k]$ code, we randomly choose $s = k + \tau$ positions, where $\tau = o(k)$ is some small positive number, e.g., $\tau \sim \log_2 k$. We wish to find an $s$-set of weight $t = 1$ in some unknown codeword of weight $w$. Let $M(n, s, w)$ denote the number of random trials needed to find such a set with a high probability $1 - e^{-n}$. Also, let $T(n, s, w)$ be the minimum number of $(n - s)$-sets needed to necessarily cover any (unknown) $w$-set. It is easy to check $[35]$ that

$$\left(\frac{n}{w}\right) / \left(\begin{array}{c}n - s \\ w\end{array}\right) \leq T(n, s, w) \leq \left(\frac{n}{w}\right) / \left(\begin{array}{c}n - s \\ w\end{array}\right) (1 + \ln \left(\begin{array}{c}n - s \\ w\end{array}\right)^{-1/2})$$  \hspace{1cm} (6)

and that $M(n, s, w) \leq T(n, s, w)n \ln n$. Below $w = 1, 2, \ldots$ $^1$This construction is analogous to pseudogenerators introduced in Ref. $[29]$. 

---

$^1$ This construction is analogous to pseudogenerators introduced in Ref. [29].
RW-algorithm performs $nT(n, s, w)$ trials of choosing $s$ random positions. Each trial gives a random $k \times s$ submatrix $G_s$ of a (random) generator matrix $G$. It is easy to verify that $G_s$ has full rank $k$ with a high probability $1 - q^{-s}$ (also, most matrices $G$ have all possible submatrices $G_k$ of rank $k-n^{1/2}$ or more). Thus, a typical $s$-set has a subset of $k$ information bits. If the current $s$-set includes such a subset, we consider $s$ vectors $(0\ldots010\ldots0)$ of weight $t = 1$ and re-encode them into the codewords of length $n$. Otherwise, we discard the $s$-set and proceed further. Algorithm stops once we obtain a codeword of weight $w$. The overall complexity has the order of $n^4 T(n, s, w)$ with the binary complexity exponent

$$F_B = H_2(\delta) - (1 - R)H_2(\delta/(1 - R)).$$

For stabilizer codes, we obtain (5) using their effective rate $R' = (1 + R)/2$.

Quantum codes with small distances $w \leq (n - k)^{1/2}$ and $s \sim nR'$ meet an exponentially tight bound

$$\log_2 T(n, s, w) \sim \log_2 \left(\frac{n}{n - w}\right)^w \sim w - w \log_2 (1 - R).$$

Exponent (5) can be further specified if codes meet the quantum GV bound (1). The corresponding exponent $F_{Bq}$ reaches its maximum $F_{\text{max}} \approx 0.22$ at $R = 0$ [Fig. 1]. By contrast, binary linear codes give exponent $F_B^s = (1 - R)\left[1 - H_2(\delta/(1 - R))\right]$ that achieves its maximum 0.12 at $R \approx 1/2$.

C. Bipartition match (BM) technique

**Proposition 3.** Code distance $\delta n$ of any quantum stabilizer code $[[n, Rn]]$ can be found with complexity exponent

$$F_{Cq} = H_4(\delta).$$

For random stabilizer codes that meet the GV bound (7),

$$F_{Cq}^* = (1 - R)/2.$$  

**Proof:** We use a sliding (“left”) window of length $s_l = \lceil n/2 \rceil$ starting in any position $i$. For any unknown vector of weight $w$, at least one position $i$ produces a window of the average weight (down to the closest integer) $v_l = \lceil w/2 \rceil$. The remaining (right) window of length $s_r = \lceil n/2 \rceil$ will have the weight $v_r = \lceil w/2 \rceil$. We calculate the syndromes of all vectors $e_l$ and $e_r$ of weights $v_l$ and $v_r$ on the left and right windows, respectively, and try to find two vectors $\{e_l, e_r\}$ that give identical syndromes, and therefore form a codeword. Clearly, each set $\{e_l\}$ and $\{e_r\}$ have size of order $L = (q - 1)^{w/2}(n/w/2)$. Finding two elements $e_l, e_r$ with equal syndromes requires complexity of order $L \log_2 L$, by sorting the elements of the combined set. Thus, finding a code vector of weight $w = \delta n$ in any classical code requires complexity of order $q^{F_{Cq}^*}$, where $F_{Cq} = H_4(\delta)/2$. For binary codes on the GB bound, $F_{Cq}^* = (1 - R)/2$. The arguments of the previous propositions then give exponents (7) and (8) for stabilizer codes.

Note that BM-technique works for any linear code, unlike two previous techniques provably valid for random codes. It is also the only technique that can be transferred to quantum codes without any performance loss. Note also that $F_{Cq}^*$ is always below the SW exponent $F_{Aq}$ and is below the RW exponent $F_{Bq}$ for very high rates. This is due to the fact that $F_{Bq} \approx 1 - R$ for $R \rightarrow 1$, and is twice the value of $F_{Cq}^*$.

D. Punctured bipartition technique

**Proposition 4.** Code distance $\delta n$ of a random quantum stabilizer code $[[n, Rn]]$ can be found with complexity exponent

$$F_{Dq} = \frac{2(1 + R)}{3 + R} H_4(\delta)$$

For random stabilizer codes that meet the GV bound (7),

$$F_{Dq}^* = (1 - R^2)/(3 + R).$$

**Proof:** We now combine the SW and BM techniques and consider a sliding window of length $s = [2nR/(1 + R)]$ that exceeds $k$ by a factor of $2/(1 + R)$. It is easy to verify that most random $[n, k]$ codes include at least one information $k$-set on any sliding $s$-window $I(i, s)$. Thus, any such window forms a punctured linear $[s, k]$ code with a smaller redundancy $s - k$. Also, any codeword of weight $w$ has the average weight $v = \lceil ws/n \rceil$ in one or more sliding windows. For simplicity, let $s$ and $v$ be even. We then use bipartition on each $s$-window and consider all vectors $e_l$ and $e_r$ of weight $v/2$ on each half of length $s/2$. The corresponding sets $\{e_l\}$ and $\{e_r\}$ have size $L_s = (q - 1)^{v/2}(s/2)^{v/2}$. We then seek all matching pairs $\{e_l, e_r\}$ that have the same syndrome $h$. Each such pair $\{e_l, e_r\}$ represents some code vector of the punctured $[s, k]$ code and is re-encoded to the full length $n$. For each $e_l = 1, 2, \ldots$, we stop the procedure once we find a re-encoded vector of weight $w$. Thus, we use $[s, k]$ punctured codes and lower BM-complexity to the order $L_s$.

However, it can be verified that some (short) syndrome $h$ of size $s - k$ can appear in many vectors $e_l$ and $e_r$ of length $s/2$, unlike the original BM-case. It turns out (35) that our choice of parameter $s$ limits the number of such combinations $e_l, e_r$ to the above order $L_s$. Thus, we have to encode all $L_s$ code
Thus, we consider the worst-case scenario in \( \Psi \) (Lemma 1 from Ref. [14]).

Lemma 1. Clusters affect disjoint sets of the parity checks. This implies that connected parts, say \( E \), can be provably applied to any code. Different parameters. In particular, the existing constructions used in quantum setting. These self-orthogonal codes represent an alternative to the BP technique used in [36] to find code.

For random codes of high rate \( R \to 1 \) that meet the GV bound, this technique gives the lowest known exponents \( F_{\text{BP}} \) (for stabilizer codes) and \( F_{\text{BP}} = R(1 - R)/(1 + R) \) (for binary codes). However, it cannot be provably applied to any linear code, unlike a simpler bipartition technique. Finally, the above propositions can be applied to a narrower class of the Calderbank-Shor-Steane (CSS) codes. Here a parity check matrix is a direct sum \( H = G_x \oplus \omega G_z \), and the commutativity condition simplifies to \( G_z T^T = 0 \). A CSS code with rank \( G_z = (n - k)/2 \) has the same effective rate \( R' = (1 + R)/2 \) since both codes include \( k' = (n - k)/2 = (n + k)/2 \) information bits. It is readily verified that CSS codes have binary complexity exponents \( F(R, \delta) \) given by expressions similar to Eqs. (2), (5), (7), (9), where one must substitute \( H_3(x) \) with \( H_2(x)/2 \).

IV. LINKED-CLUSTER TECHNIQUE

Let \( \Psi(s, \ell) \) be an ensemble of regular \((s, \ell)\) LDPC codes, in which every column and every row of matrix \( H \) has weight \( s \) and \( \ell \) respectively. The following technique is designed as an alternative to the BP technique used in [36] to find code distance. First, note that with quantum codes, BP can yield decoding failures [23], while our setting requires error-free guarantee. The second, more important, reason is that we consider very specific, self-orthogonal LDPC codes that can be used in quantum setting. These self-orthogonal codes represent very atypical examples of \( \Psi(s, \ell) \) and can have drastically different parameters. In particular, the existing constructions of such codes have low distance \( d \), where \( \log d \sim (\log n)/2 \), whereas a typical \((s, \ell)\)-code has a linearly growing distance. Thus, we consider the worst-case scenario in \( \Psi(s, \ell) \), which can be provably applied to any code.

For an \((s, \ell)\)-code, we represent all \((qu)\) bits as nodes of a graph \( G \) with vertex set \( V(G) \) and connect two nodes iff there is a parity check that includes both positions. A codeword \( \epsilon \) is defined by its support \( E \subseteq V(G) \) and induces the subgraph \( G(E) \) that forms one or more clusters and has no edges outside of \( G(E) \). Generally, we will make no distinction between the set \( E \) and the corresponding subgraph. Note that disconnected clusters affect disjoint sets of the parity checks. This implies

Lemma 1 (Lemma 1 from Ref. [14]). A minimum-weight code word of a q-ary linear code forms a linked cluster on \( G \).

Proof: Let a minimum-weight support \( E \) include disconnected parts, say \( E_1 \) and \( E_2 \). These parts satisfy different parity checks. Then vectors generated by \( E_1 \) and \( E_2 \) belong to our code and have smaller complexity. Contradiction.

Linked-cluster algorithm. The following breadth-first algorithm inspects all fully-linked clusters of a given weight \( w = \eta \). Let us assume that \( j = 0, 1, \ldots, n - 1 \) is the starting position in the support \( E \) of an unknown codeword of weight \( w \). Position \( j \) belongs to some \( s \) parity checks which form the list \( \eta = \{h_1, \ldots, h_s\} \). To satisfy the parity-check \( h_1 \), we arbitrarily select some (odd) number \( v_1 \) of the remaining \( \ell - 1 \) parity-check positions of \( h_1 \). These \( v_1 \) positions are now included in the current support \( \bar{E} \). Any time a new position is selected, we also append the list \( \eta \) with the new parity checks which include this position. We then proceed with the subsequent parity-checks \( h_2, h_3, \ldots \) as follows. Let a check \( h_i \) overlap with some of the parity checks \( h_1, \ldots, h_{i-1} \) in \( a_1 \leq \ell - 1 \) positions, and let \( b_i \) be the number of 1s selected in these \( a_i \) positions. Then \( h_i \) can use only the remaining \( \ell - a_i \) positions to pick up some \( v_i \equiv b_i \) (mod 2) positions. If \( b_i \) is odd, the algorithm adds some \( v_i \in \{1, 3, \ldots\} \) positions from \( h_i \), but (temporarily) skips this check if \( b_i \) is even. This parity check \( h_i \) can be re-processed in some later step \( p \) as a parity check \( h_p \) if the corresponding number \( b_p \) is odd. The process is stopped once we add \( v = w - 1 \) positions. The result is a binary codeword with support \( E \) if all processed odd-type checks are satisfied and all unprocessed checks have even overlap with \( E \). At this point, adding some \( v_i = 2, 4, \ldots \) symbols in any even-type check can only increase the weight of a codeword. For a q-ary code, we perform summation over \( v_i = 1, 2, \ldots, n \) and need to check the rank of a matrix formed by the corresponding \( v \) columns of the check matrix. For a quantum stabilizer code, we also need to verify that any obtained codeword is linearly independent from rows of \( H \).

At step \( i \), there are \( \ell^{v_i} \) ways to select \( v_i \) positions. Thus, the total number of choices \( N_v \) to select \( v \) positions is

\[
N_v \leq \sum_{m \geq 1} \sum_{v_i \in \{1, 3, \ldots\}} \delta_{v_1 v_1 + \ldots + v_m} \prod_{i=1}^{m} \ell^{v_i - 1}
\]

which in turn is bounded by

\[
S_v(\ell) \equiv \sum_{m \geq 1} \sum_{v_i \in \{1, 3, \ldots\}} \delta_{v_1 v_1 + \ldots + v_m} \prod_{i=1}^{m} \ell^{v_i - 1}
\]

Here \( \delta_{a,b} \) is the Kronecker symbol, and \( m \) is the number of terms in the decomposition \( v = v_1 + \ldots + v_m \).

To estimate \( S_v(\ell) \), introduce the generating function \( g_v(z) = \sum_v S_v(\ell) z^v \). Easy summation gives for \( q = 2 \):

\[
g_v(z) = (1 - f_\ell(z))^{-1}, \quad f_\ell(z) = \frac{(1+z)^{\ell-1} - (1-z)^{\ell-1}}{2}
\]

Finally, we use the contour integration of \( g_v(z) \) to find the coefficients \( S_v(\ell) \). Let \( \gamma = \sinh^{-1}(1) \approx 1.135 \) in the case of a binary code, and \( \gamma = 1/\ln 2 \approx 1.443 \) in the case of a q-ary code. We have:

Proposition 5. A codeword of weight \( \delta n \) in a \((s, \ell)\)-code can be found with complexity exponent \( F_{1,\ell} = \delta \log_2(\gamma \ell (\ell - 1)) \), where \( \gamma \in (1, \ell) \) grows monotonically with \( \ell \).

More precise estimates of \( S_v(\ell) \) also give specific numbers \( \gamma \), which can be important for small values of \( \ell \). Finally
note that while the cluster technique has high complexity for large $\ell$ and $\delta$, its exponent $F_{LC}$ is linear in the relative distance $\delta$. In comparison, deterministic techniques of Sec. III give the higher exponents $F \propto \delta \log(1/\delta)$ in this limit. All known quantum LDPC codes with limited-weight stabilizer generators have $\delta \propto n^{-1/2}$, and the linked-cluster technique gives the lowest complexity for these codes. Note that the RW technique also gives a linear in $\delta$ exponent $F_{RW}$ that is bounded by $\delta - \delta \log_2(1 - R)$. Our cluster technique still lowers this exponent $F_{RW}$ for high code rates $R \geq 1 - 2[\gamma (\ell - 1)]^{-1}$.

V. CONCLUSION

In this paper, we considered different techniques of finding code distances of stabilizer quantum codes. For sparse quantum LDPC codes, we proposed a new cluster-based technique. This technique reduces complexity exponents of the existing non-probabilistic algorithms for codes with sufficiently small relative distances. In particular, this is the case for all known families of quantum LDPC codes that have distances of order $n^{1/2}$ or less. Cluster-based technique also beats the probabilistic random-window technique for high-rate codes.

Acknowledgment. This work was supported in part by the U.S. Army Research Office under Grant No. W911NF-11-1-0027, and by the NSF under Grant No. 1018935.

REFERENCES

[1] P. W. Shor, “Scheme for reducing decoherence in quantum computer memory,” Phys. Rev. A, vol. 52, p. R2493, 1995. [Online]. Available: http://link.aps.org/abstract/PRA/v52/pR2493
[2] E. Knill and R. Laflamme, “Theory of quantum error-correcting codes,” Phys. Rev. A, vol. 55, p. 900–911, 1997. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.55.900
[3] C. Bennett, D. DiVincenzo, J. Smolin, and W. Wootters, “Mixed state entanglement and quantum error correction,” Phys. Rev. A, vol. 54, p. 3824, 1996. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.54.3824
[4] A. Y. Kitaev, “Fault-tolerant quantum computation by anyons,” Ann. Phys., vol. 303, p. 2, 2003. [Online]. Available: http://arxiv.org/abs/quant-ph/9707021
[5] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, “Topological quantum memory,” J. Math. Phys., vol. 43, p. 4452, 2002. [Online]. Available: http://dx.doi.org/10.1063/1.1499754
[6] H. Bombin and M. A. Martin-Delgado, “Topological quantum distillation,” Phys. Rev. Lett., vol. 97, p. 180501, Oct. 2006. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevLett.97.180501
[7] ———, “Optimal resources for topological two-dimensional stabilizer codes: Comparative study,” Phys. Rev. A, vol. 76, no. 1, p. 012305, Jul 2007.
[8] ———, “Homological error correction: Classical and quantum codes,” Journal of Mathematical Physics, vol. 48, no. 5, p. 052105, 2007. [Online]. Available: http://scitation.aip.org/content/aip/journal/jmp/48/5/10.1063/1.2731356
[9] R. Raussendorf and J. Harrington, “Fault-tolerant quantum computation with high threshold in two dimensions,” Phys. Rev. Lett., vol. 98, p. 190504, 2007. [Online]. Available: http://link.aps.org/abstract/PRL/v98/e190504
[10] H. G. Katzgraber, H. Bombin, and M. A. Martin-Delgado, “Error threshold for color codes and random three-body ising models,” Phys. Rev. Lett., vol. 103, p. 090501, Aug 2009. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevLett.103.090501
[11] S. Bravyi, D. Poulin, and B. Terhal, “Tradeoffs for reliable quantum information storage in 2D systems,” Phys. Rev. Lett., vol. 104, p. 050503, Feb 2010. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevLett.104.050503