

Decoding topological *XYZ*^{2} codes with reinforcement learning based on attention mechanisms

Qing-Hui Chen(陈庆辉), Yu-Xin Ji(姬宇欣), Ke-Han Wang(王柯涵), Hong-Yang Ma(马鸿洋), Nai-Hua Ji(纪乃华)

Citation: Chin. Phys. B, 2024, 33 (**6**): 060314. DOI: 10.1088/1674-1056/ad342b
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Recurrent neural network decoding of rotated surface codes based on distributed strategy

Fan Li(李帆), Ao-Qing Li(李熬庆), Qi-Di Gan(甘启迪), and Hong-Yang Ma(马鸿洋)
Chin. Phys. B, 2024, 33 (**4**): 040307. DOI: 10.1088/1674-1056/ad2bef

Performance of entanglement-assisted quantum codes with noisy ebits over asymmetric and memory channels

Ji-Hao Fan(樊继豪), Pei-Wen Xia(夏沛文), Di-Kang Dai(戴迪康), and Yi-Xiao Chen(陈一骁)
Chin. Phys. B, 2023, 32 (**12**): 120304. DOI: 10.1088/1674-1056/acf492

Approximate error correction scheme for three-dimensional surface codes based reinforcement learning

Ying-Jie Qu(曲英杰), Zhao Chen(陈钊), Wei-Jie Wang(王伟杰), and Hong-Yang Ma(马鸿洋)
Chin. Phys. B, 2023, 32 (**10**): 100307. DOI: 10.1088/1674-1056/acd8a9

An overview of quantum error mitigation formulas

Dayue Qin(秦大粤), Xiaosi Xu(徐晓思), and Ying Li(李颖)
Chin. Phys. B, 2022, 31 (**9**): 090306. DOI: 10.1088/1674-1056/ac7b1e

Determination of quantum toric error correction code threshold using convolutional neural network decoders

Hao-Wen Wang(王浩文), Yun-Jia Xue(薛韵佳), Yu-Lin Ma(马玉林), Nan Hua(华南), and Hong-Yang Ma(马鸿洋)
Chin. Phys. B, 2022, 31 (**1**): 010303. DOI: 10.1088/1674-1056/ac11e3

Decoding topological XYZ^2 codes with reinforcement learning based on attention mechanisms

Qing-Hui Chen(陈庆辉)[1], Yu-Xin Ji(姬宇欣)[1], Ke-Han Wang(王柯涵)[2], Hong-Yang Ma(马鸿洋)[2], and Nai-Hua Ji(纪乃华)[1,†]

[1] *School of Information and Control Engineering, Qingdao University of Technology, Qingdao* 266033, *China*
[2] *School of Sciences, Qingdao University of Technology, Qingdao* 266033, *China*

(Received 14 December 2023; revised manuscript received 18 February 2024; accepted manuscript online 15 March 2024)

Quantum error correction, a technique that relies on the principle of redundancy to encode logical information into additional qubits to better protect the system from noise, is necessary to design a viable quantum computer. For this new topological stabilizer code-XYZ^2 code defined on the cellular lattice, it is implemented on a hexagonal lattice of qubits and it encodes the logical qubits with the help of stabilizer measurements of weight six and weight two. However topological stabilizer codes in cellular lattice quantum systems suffer from the detrimental effects of noise due to interaction with the environment. Several decoding approaches have been proposed to address this problem. Here, we propose the use of a state-attention based reinforcement learning decoder to decode XYZ^2 codes, which enables the decoder to more accurately focus on the information related to the current decoding position, and the error correction accuracy of our reinforcement learning decoder model under the optimisation conditions can reach 83.27% under the depolarizing noise model, and we have measured thresholds of 0.18856 and 0.19043 for XYZ^2 codes at code spacing of 3–7 and 7–11, respectively. our study provides directions and ideas for applications of decoding schemes combining reinforcement learning attention mechanisms to other topological quantum error-correcting codes.

Keywords: quantum error correction, topological quantum stabilizer code, reinforcement learning, attention mechanism

PACS: 03.67.–a, 87.64.Aa, 03.67.Pp **DOI:** 10.1088/1674-1056/ad342b

1. Introduction

In recent years quantum computing and quantum information have been developing rapidly,[1–4] and quantum error correction, as a powerful and necessary quantum computing method, has been searching for a fast and accurate method of correcting qubits that protects quantum computation and storage as well as protects quantum information from being lost.[5,6] The modern theory of error-correcting codes in quantum information has become a hot research issue in the field of quantum error correction (QEC) codes, and topological stabilizer codes[7–9] are undoubtedly the most popular class of quantum error-correcting codes under study. Quantum topological stabilizer codes are physically fault-tolerant to weakly quasi-local perturbations, and thus also exhibit high fault tolerance thresholds and relatively low overheads. Quantum error correcting codes based on the Kitaev honeycomb model are topological quantum error correcting codes[10–12] that utilize spin and time-reversal symmetries on a hexagonal lattice to protect quantum information. XYZ^2 can be traced back to this model,[13,14] which is a typical model of topological ordering because the product of the linking operators around each hexagonal vignette of the lattice is exchanged with the Hamiltonian quantities, thus presenting a set of conserved quantities that can be used as stabilizers for error-correcting codes. Due to the fragility of quantum information and in order to suppress errors in logical qubits, a future general-purpose quantum computer could diagnose syndromes based on the logical qubits of the stabilizer.[15]

A key subroutine of QEC is decoding.[16,17] A classical decoding process is the error detection and correction of relevant information in a quantum system due to environment and noise, etc., so that the quantum state information is error-free. To further prevent error propagation and logic failures, the decoder provides a set of recovery operations to correct specific syndrome errors.[18] Thus, the design of the decoder in particular needs to be improved. Recently, there has been an increase in the development of machine learning based decoders. Machine learning based decoding algorithms can achieve similar decoding performance as the classical decoding algorithms.[19–21] In addition, machine learning can be used to compute the error correction thresholds in quantum error correction and requires less execution time. Deep learning has also received attention from various disciplines and has a wide range of applications in quantum error correction. Machine learning can be used to build decoder frameworks, and deep learning can decode surface codes.[22,23] However, the main problem with such decoders is that during the training of a neural network, as the size of the system increases, more training samples need to be collected to train the neural network in order to obtain a decoding performance similar to that of classical algorithms. This makes training more difficult

[†]Corresponding author. E-mail: 13964863452@126.com

© 2024 Chinese Physical Society and IOP Publishing Ltd

and slow. The development of reinforcement learning (RL) combined with deep learning can solve such problems.[24–26]

There are many decoders designed based on surface and color codes,[27] whereas decoders based on XYZ^2 codes are rarely involved. The effective weight and degeneracy (EWD) decoders have been proposed with XYZ^2 thresholds of roughly 18%.[28–30] It is based on Monte Carlo sampling of misconfigurations using the Metropolis algorithm. Essentially an improvement on the traditional Markov chain Monte Carlo method, this decoder has improved performance but requires larger data samples. In this paper, we made a new attempt, we chose to introduce a reinforcement learning module using an attention mechanism based on an encoder-decoder architecture.[31] Then the reward in reinforcement learning is combined with the attention mechanism to make the agent more able to pay attention to the valuable input and output states. It is illustrated that the current work performs faster analysis and decision making than existing work, it helps to improve the reliability and processing efficiency of quantum error correcting codes. Through the analysis of experimental simulation data, it is found that the trained model in this study has higher error correction accuracy than the general reinforcement learning model. The error correction code in this study will have a higher threshold and the decoding model will have a higher accuracy.

The rest of the paper is structured as follows. In Section 2 we provide background knowledge on XYZ^2 codes, reinforcement learning, and attentional mechanisms. In Section 3 we describe the decoding strategy of this study. In Section 4 we present the algorithm designed for XYZ^2 codes. In Section 5 we discuss the error correction performance analysis of the decoder. Section 6 summarizes the conclusions.

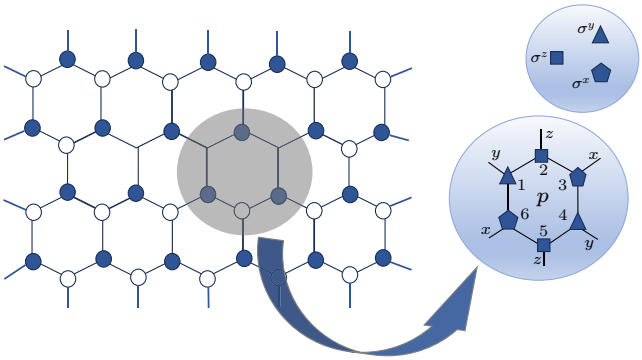


Fig. 1. Numbers 1–6 mark the lattice positions on a single hexagonal plaquette p, with p being the label of the plaquette operator, i.e., a closed-loop operator around each hexagon of the lattice. The honeycomb lattice interacts with the plaquette operator W_p, $p = -1$, and the Kitaev on the honeycomb lattice.

2. Background

2.1. The XYZ^2 code

The discovery and applications of topological properties compared to standard QEC codes provide many ideas for further research. The Kitaev cellular model has had a great impact in the field of new topologies, and continues to provide new ideas for the sustainable development of topological properties in the quantum domain. Inspired by this model, a topological stabilizer code with plaquette and link operator stabilizers, the "XYZ^2" code, is defined on a cellular lattice with a specific boundary. The XYZ^2 code is also a type of error-correcting code that needs to be studied in depth in topological codes. The Kitave honeycomb model defines that quantum bits are generally on the honeycomb lattice, for which a $(XYZXYZ)$ plate stabilizer with weight 6 is utilized on a planar hexagonal lattice consisting of $2d^2$ quantum bits and weight-two (XX) link stabilizers, where the vertical link stabilizers can also be YY or ZZ. In this paper, we take the XX vertical link stabilizer as an example.[32–34] Quantum bits are associated with the edges of a two-dimensional lattice, while stabilizer operators correspond to faces and points. XYZ^2 is defined in a hexagonal lattice p, as shown in Fig. 2(a), and canonical stabilizers are formulated on the hexagonal lattice, in addition to the two kinds of stabilizers just introduced, we also newly add the semiperipheral boundaries and introduce semiperipheral boundary stabilizers that stabilize a logical quantum bit using the stabilizers with a weight of two, denoted as three x, y, and z, with a general code spacing of d at the boundary at this point, to enable error correction by using the link as the canonical operator of the error-correcting code of the subsystem code. The associated error syndrome can be found by measuring the state of the plaquettees operator and the link operator operator to detect their errors periodically.[35] For the cellular dot matrix, as shown in Fig. 2, the link operator operator can be expressed as

$$K_l = \sigma_j^{\alpha}\sigma_k^{\alpha}, \tag{1}$$

where l is denoted as a connection acting on two neighboring qubits, and the representation type is $\alpha \in \{x, y, z\}$, represented by three Pauli matrices x, y, and z, respectively. If the bonds j, k are of type γ, denoting the exchange interactions occurring between sites j, k connected by γ chains. Then we have

$$K_{j,k}^{\gamma} = \sigma_j^{\gamma}\sigma_k^{\gamma}. \tag{2}$$

The plaquette operator should be defined as

$$W_p = \sigma_1^x\sigma_2^y\sigma_3^z\sigma_4^x\sigma_5^y\sigma_6^z, \tag{3}$$

where σ_i^a is the associated Pauli operator at i. The lattice is of the hexagonal plate p where the numbers 1–6 on a single hexagonal plate p marking the lattice positions (see Fig. 1) are closed-loop operators around each hexagon of the lattice. Each plaquette (enclosing surface) operator is exchangeable with all link operators and their products. Therefore, these plaquette operators are also exchangeable with each other. Here each

stabilization operator is a product of Pauli matrices acting on different qubits, which can again be expressed as

$$\hat{W}_p = \prod_{j\in p} K^{\gamma}_{j,j+1}. \tag{4}$$

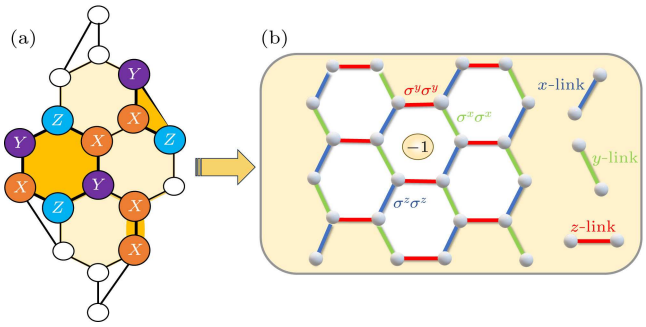


Fig. 2. (a) The hexagonal lattice structure of a novel honeycomb lattice topological error correcting code, the XYZ^2 code, with the parts that are deepened in color and enlarged by bits being each of the three types of stabilizers mentioned earlier. (b) The honeycomb lattice with the plaquette operator W_p, $p = -1$, interacting with the Kitaev on the honeycomb lattice by tiling it down as shown. The model consists of three types of link operators, depending on their orientation; we call them "x-links", "y-links", and "z-links". This definition can be extended to the whole lattice using translation invariance.

2.2. Attention mechanism

Attention mechanisms have also started to be applied to deep learning in recent years.[36,37] Attentional mechanisms help reinforcement learning models to obtain more accurate state information. Here the model architecture of the attention mechanism we use the encoder-decoder framework, which consists of an encoder and a decoder,[38] the mechanism is trained to learn the decoding policy function, the Q-value function, or the policy gradient function in the case of combining with reinforcement learning algorithms. These functions can help the decoder to make error correction decisions based on the current system state and attention weights.

In reinforcement learning, an autonomous agent interacts with the environment through its own experience,[39] observes the state of the environment to take action, and learns its own behaviors with the help of feedback from the environment to maximize the cumulative rewards. These behaviors tend to become more optimal as the agent gains experience.[39,40] This can be solved by employing a Markov decision process[41] consisting of five different entities: state, action, reward, policy, and value. Reinforcement learning can be represented by a quaternion $\langle S, A, P, R\rangle$, where S is the set of states, A is the set of actions, P is the state transfer function, and R is the reward function. The current state of the environment is determined by the current samples in the training or test set. At a certain moment t, the agent receives the first sample, and the intelligent chooses an action, $a \in A$, from the action set based on the current state of the environment, $s_t \in s$, and a policy. After completing the action it sends an action signal a_t (reward or punishment) to interact with the environment, and based on the environment, the agent keeps updating its own policy π with the information of the current state s_{t+1} and the reward $r_t = r(s_t, a_t)$ returned from s_t in the order of the samples in the training dataset, the information is constantly updated to find an optimal policy that accumulates in order to obtain the maximized reward, where the cumulative reward obtained by accumulating is

$$R_t = \sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}). \tag{5}$$

The discount factor is denoted by γ, where $\gamma \in (0,1)$. It is used to quantify how people want to assess immediate and subsequent rewards.[42] By using a measure of cumulative rewards, the value of any given state depends not only on the immediate rewards of the particular policy followed by that state, but also on the expected future rewards.

3. Models and algorithms

3.1. Exploration of the XYZ^2 code

For a dichotomous graph lattice point where all planes contain an even number of edges with eigenvalues ± 1, we refer to this as the zero $(+1)$ or π (-1) flux. It is easy to verify that all plaquette operators are exchangeable with each other and with Hamiltonian operators, and here the product of link operators around each hexagon is exchanged with Hamiltonian quantities to give a set of conserved quantities that can be used as stabilizers for error-correcting codes. It is shown that all plaquette operators are linked to each other and are exchangeable with Hamiltonian quantities.

A cell in the lattice contains one vertex of each type, which is categorized into x-links, y-links, and z-links in the case of this paper, and the Hamiltonian reads

$$H = -J_x \sum_{x\text{-links}} \sigma^x_j \sigma^x_k - J_y \sum_{y\text{-links}} \sigma^y_j \sigma^y_k - J_z \sum_{z\text{-links}} \sigma^z_j \sigma^z_k, \tag{6}$$

where J_x, J_y, and J_z are model parameters. Let us introduce a special notation for the individual terms in the Hamiltonian quantity:

$$K_{jk} = \begin{cases} \sigma^x_j \sigma^x_k, & \text{if } (j,k) \text{ is an } x\text{-link;} \\ \sigma^x_j \sigma^y_k, & \text{if } (j,k) \text{ is an } y\text{-link;} \\ \sigma^x_j \sigma^z_k, & \text{if } (j,k) \text{ is an } z\text{-link.} \end{cases} \tag{7}$$

A closer look reveals that one of these operators will be interchangeable with the W_p operator in Eq. (3) associated with the hexagonal lattice of Fig. 2. These operators W_p are associated with the hexagonal lattice slab, and in order to find the eigenstates of the Hamiltonian quantity, we first divide the full Hilbert space $\mathcal{L}$ into the sector-eigenstates of W_p, which are also the invariant subspaces of H. This can be expressed as

$$\mathcal{L} = \bigoplus_{w_1,\ldots,w_m} \mathcal{L}_{w_1,\ldots,w_m}, \tag{8}$$

where m is the number of plaquettes, and each operator W_p has eigenvalues $+1$ and -1, so that each going up corresponds to a choice of $W_p = \pm 1$ for each plaquette p. We then need to solve for the eigenvalues of the Hamiltonian restricted to a particular sector $\mathcal{L}_{w_1,\ldots,w_m}$.

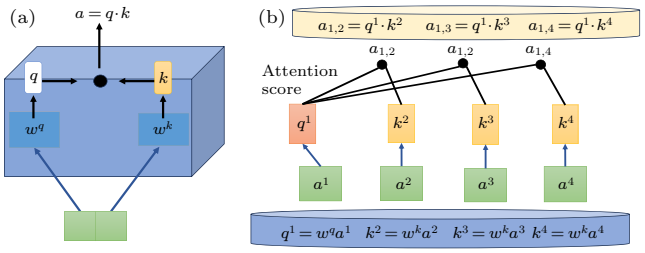


Fig. 3. (a) Based on dot product calculation, the correlation between two state vectors is calculated. (b) Dot product the state vector of q^1 and the target vectors of each of the four keys to obtain the four correlation values. The correlation values are updated by the neural network and then normalized by a softmax layer. These values are also referred to as "attention scores", which in turn affect the current state processing and decision making.

3.2. State attention mechanism

Attention networks[43] use feature vectors as inputs and outputs for attention values, and an attention function maps queries and key-value pair tuples to outputs. This step serves to give a weight distribution based on the information of the query and the relevance of the key-value pairs, which is used to guide the decoder in focusing its attention on different locations during the decoding process, where the change in the A_k value given by the attention value network depends on the output of the learning agent o_k,

$$\frac{\partial L}{\partial f_{ak}(S_t)} = \left[\frac{\hat{y}-y}{\hat{y}-\hat{y}^2}\right] \cdot \underbrace{f_d'(O_t) \cdot o_k}_{} \cdot \left[\frac{(\sum e^{A_k}) \cdot (e^{A_k}) - (e^{A_k})^2}{(\sum e^{A_k})^2}\right]. \tag{9}$$

This allows the decoder to pay more attention to the information related to the current decoding position and improves the accuracy of the decoding. Here, an auxiliary function with corresponding key inputs calculates the weights assigned to each value. Finally the weighted sum of the input values is output. The attention mechanism is simple and can give the model greater discriminative power.

The attention mechanism used by researchers in other directions is generally the self-attention mechanism, but in the quantum error correction direction we use the type of state attention mechanism that introduces additional state information, the self-attention mechanism only considers the correlation between different positions in the input sequence, in order to make the agent more able to pay attention to the valuable states of the input state, and more accurately pay attention to the information related to the current decoding position. The state attention mechanism generates a query vector based on the decoded information of the decoder state, which is used to guide the decoder's attention allocation during the decoding process. Typically, the attention mechanism computes the attention weights and combines them with the encoder's output feature vectors in a weighted manner to provide more targeted inputs to the reinforcement learning model [see Fig. 4(c)].

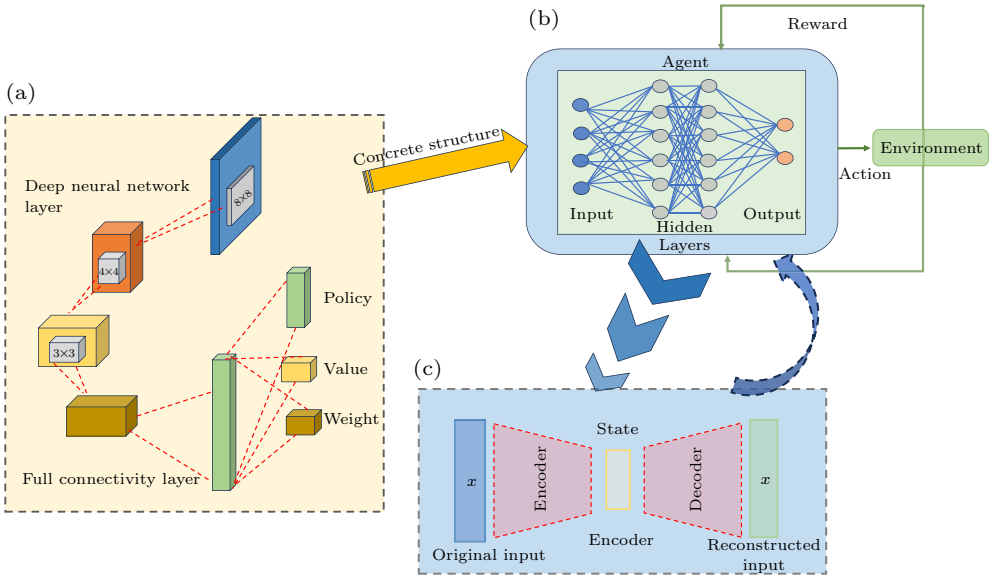


Fig. 4. Structure of the deep-Q-network and encoder-decoder architecture.

3.3. Network architecture under the influence of attentional mechanisms

The hidden layer in the network architecture in Fig. 4(a) utilizes three deep neural network layers and one fully connected layer that will depolarize the syndrome obtained under the model noise, and in our case the state will correspond to a medium syndrome. The inputs to the network architecture are three deep neural network layers with filter sizes of 8, 4, and 3. The network has three outputs, which output policies, values, and attention weights for the DQN algorithm based on the

attention mechanism. The agent utilizes a deep convolutional neural network or Q-network to approximate the Q-value of the action for a given syndrome.

The syndrome inputs and the three outputs of the network are then used to train the encoder and decoder model, which is divided as follows [see Fig. 4(c)]: the encoder stage mainly affects the assignment of weights, which are computed by weighting the different erroneous parts of the input sequence. In this phase, the attention mechanism helps to focus on the critical error bit locations and enhances the focus on the error qubits. The decoder is used to perform the nonlinear mapping and to output the information about the relevant error quantum states for input to the reinforcement learning decoder. Additionally, the attention mechanism allows the decoder to focus on input locations that are relevant to the current output, thereby better capturing the correspondence between inputs and outputs, with implications for policy generation and value assessment. The reinforcement learning decoder is trained to augment the results learnt by the attention mechanism to increase the probability of finding the location of the error syndrome, thus reinforcing the accuracy and efficiency of error correction.

4. Algorithmic process

4.1. Construction of the decoder

The action-value Q-function encodes the discounted value of moving a corrector to an adjacent position on the square grid according to the set of corrector subsets on the hexagonal ring. All stabilizers are interchangeable and have an eigenvalue of ± 1. Errors are detected by measuring the stabilizer generator, and the bubblegum error is commuted with certain stabilizers, measuring these stabilizers returns an eigenvalue of $+1$ if no error occurs, or -1 if it does. The advantage of the XYZ^2 code is that fewer auxiliary quantum bits per data qubit are used to perform stabilizer measurements,[44] and that local errors can be detected and corrected by encoding the quantum information in nonlocal encoding quantum information in a non-local manner to detect and correct local errors.

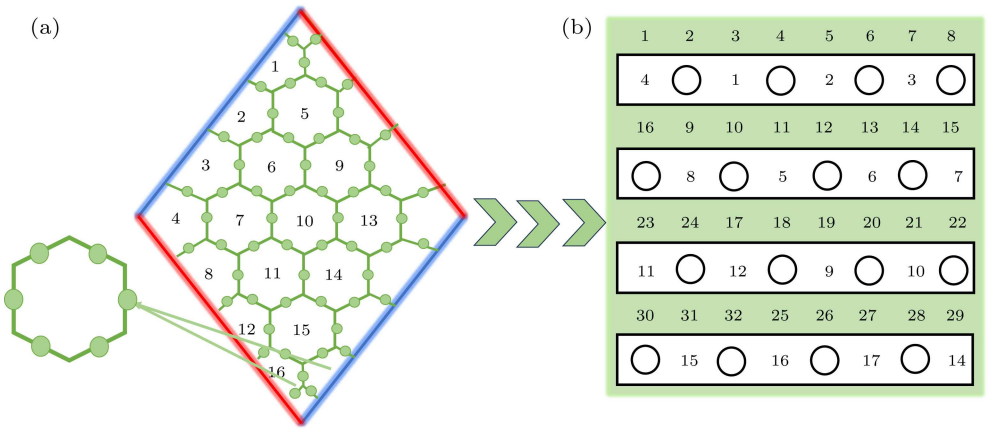


Fig. 5. (a) Schematic diagram of hexagonal lattice of the XYZ^2 code embedded in the ring surface with code spacing 4. There are 9 patches, and there are 7 pairs of two-by-two combinations of half-face patches, so there are a total of 32 vertices with 48 physical qubits. (b) The transformed square grid corresponding to (a), with the vertices and the plalette operator labeled from left to right and from top to bottom, in order, as an 8×8 image. Here ◯ has no meaning, just for spatial validity.

A syndrome is a measurement used to detect and diagnose quantum errors. A reinforcement learning decoder can extract information from the changes in the syndrome data to adapt its own strategy to maximize the effect of error correction. Before designing the reinforcement learning decoder, we must map the XYZ^2 stabilizer code with a honeycomb model-like shape to a ring surface code first, and map its hexagonal dot matrix to a square programming framework, and in Ref. [45] the authors provided directions and ideas for our mapping approach, and the present paper gives the results directly after mapping. Figure 5(a) shows the XYZ^2 stabilizer code map of the embedding that we converted into a suitable deep neural network.

In a deep neural network model, the output layer is a set of Q-values, and to approximate the Q-values for a given syndrome activity, the agent uses a Q-network,[46] where these Q-values represent the decoder's expected reward or utility for each action. By selecting the action with the highest Q-value, the decoder can decide which action to perform to recover from the error.

4.2. Emulation of XYZ^2 codes decoder

The decoder proposed in this paper is a deep neural network based agent that uses a reinforcement learning approach combined with an attentional mechanism to improve the efficiency of finding the best error syndrome on XYZ^2 codes. The decoder process is to provide the algorithm as an input value, where the estimated rewards are measured iteratively in Q-learning and stored in a tabular form for each state and each potential action in that state. As the number of states and ac-

tions increases, we will have difficulty in storing and updating the table. Thus, the concept of deep-Q-network (DQN)[47,48] is introduced, where the states and actions are treated as inputs to the neural network, and then the neural network is analyzed to obtain the Q-values of the actions.

This makes it easier to store and update each Q-value for each iteration of the Q-learning algorithm. $Q(s,a;\theta)$ denotes the average expected reward when taking action a in state s, where θ denotes the trainable weight of the network. Figure 4(b) illustrates the structure and process of the DQN. Firstly, data needs to be collected as a training dataset, then the states are used as inputs to the DQN and the predicted reward values for all possible actions are output. Based on the predicted reward values for the actions, the agent program selects and executes the actions. Based on the executed action, the agent program moves from the current state to the next state. The environment validates the action and provides feedback to the agent program, the rewards are given based on the executed action. Strong reward will be fed back if the action performed is the desired error correction; otherwise, weak. The reward from the environment is used to update the weights of the DNN. As a result, the performance of the DNN will improve over time. The optimal Q-function is denoted by $Q^*(s,a;\theta)$:

$$Q^*(s,a;\theta)=\max_{\pi}\mathbb{E}[r_t+\gamma\cdot r_{t+1}+\gamma^2\cdot r_{t+2}+\cdots\,|\,s_t=s,a_t=a,\pi], \tag{10}$$

where γ_t and γ denote the reward t and discount factor of time, respectively. The maximum reward value is selected among all possible strategies as shown in Eq. (11). The strategy π of the DQN varies with the parameter θ. The expected reward $Q(s,a;\theta)$ is computed iteratively until its value converges as given by Bellman's equation.[49] This is given by

$$Q(s,a;\theta)=r+\gamma\cdot\max_{a'}Q(s',a';\theta'). \tag{11}$$

To improve the exploration of the policy, we typically perform Q-learning using an ε-greedy policy that takes the optimal action $(1-\varepsilon)$ with probability on the estimation of $Q(s,a)$ and chooses an action randomly with probability ε. In addition, we gradually decrease the value of ε, reducing it linearly to a minimum of $\varepsilon=0.1$.

Combining it with an attention mechanism, the intelligence pays more attention to valuable states during the decoding phase, at each time step t. A weighting network w_ϑ is introduced, which assigns different weights to the rewards under each moment, at which point the cumulative reward is then expressed as

$$R_t=\sum_{k=0}^{\infty}\gamma^k w_\vartheta(s_{t+k})r(s_{t+k},a_{t+k}). \tag{12}$$

The specific process of reinforcement learning decoding based on the attention mechanism is as follows: the attention mechanism defines a query and a key value that represents the state vector of all qubits of the query and correlates it with the key value of the qubit corresponding to the error syndrome. The reward for the current state is obtained by inputting the current action, e.g., the state information of the qubits, the type of error flip, and the information of the error correction operation, and updating the attentional weights, and the feedback obtained, e.g., the current state, the action, and the reward is used for the next step. During the error correction process, the decoder selects the appropriate attention allocation strategy based on the information that has been decoded and the state of the current decoding position as the history state. By inputting the current state s_t, here s_t is used as an input to the attention value decision network, and its output is represented by $f_a(s_t)$, where f_a is a nonlinear transformation of s_t. The model learns to relate the attention value to the knowledge of the agent in the network, and the better the performance of the agent, the higher the attention value will be. Then we get the attention value of the intelligence as

$$\alpha_k=\frac{e^{f_{ak}(s_t)}}{\sum_k e^{f_{ak}(s_t)}}. \tag{13}$$

Updating the weights of the attention network and continuously optimizing the correlation between the query state and the historical state to improve the efficiency of finding the error syndrome. The specific process is shown in Fig. 3. Each layer outputs the correlation coefficients between state vectors represented by neurons, which are specifically obtained by dot product operation. A comprehensive average correlation coefficient is obtained by weighted average of different state vector correlation coefficients. After a series of neural network layer processing,[43] the output correlation state information and the weighted average are combined together and fed into our reinforcement learning decoder.

The attention value computed by the agent is attributed to the weighted feature $O_t=\sum_k\alpha_k\cdot o_k$ of the computational state s_t. The decoder then decides the state s_t network and is represented by f_d with applying a nonlinear transformation to O_t in the form of a decision neuron $\hat{y}=f_d(O_t)$, where $\hat{y}$ is a softmax output indicating the probability of an incorrect type of state, the loss function of the agent's DQN network is defined as

$$L=[r+\gamma\cdot\max_{a'}Q(s,a';\theta')-Q(s,a;\theta)]^2. \tag{14}$$

The loss function of the agent's deep Q-network is defined successfully as the agent uses the above loss function to update its own deep Q-network parameters. Reinforcement learning combined with an attention mechanism is used to optimize the observation of syndromes for XYZ^2 codes.

Three independent samples are selected for training set, validation set and test set. Here 50% of the samples are training set and 25% of the samples are validation and test set, all three parts are randomly selected from the samples. The training set is used to estimate the parameters of the model algorithm to learn the error correction policy and to optimize the

model on the training set. The validation set is used to determine the network structure or parameters that control the complexity of the model, and the test set is used to test the performance of the optimal model that is finally selected.

5. Error correction performance analysis

Depolarizing noise is a general error model widely used to model quantum error correcting codes. The error model in this paper takes the depolarizing noise model as an example to train the decoder. Under the training of the decoder in this paper, the data plotted in Fig. 6 are obtained, and the test accuracy can closely follow the training accuracy, and the training accuracy can reach 83.27%.

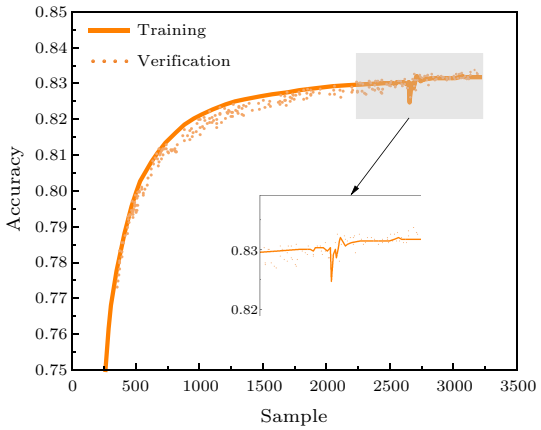


Fig. 6. Successful decoding agent training and testing accuracy versus sample size. The horizontal axis indicates the number of samples and the vertical axis indicates the training accuracy. The achieved and dashed lines under the same color indicate the accuracy of the training and test sets, respectively. Local zoom plots are set to allow for observation of the data.

In quantum error correction, the distance d of the code quantitatively determines the number of single qubit errors that can be corrected in the absence of logic errors. A measure of decoding efficiency usually considers the error threshold. The threshold can be viewed as a threshold that protects the probability that an error will be generated during the decoding of an error-correcting code. Codes with high error thresholds are favorable. The logic failure rate depends strongly on the distance of codes below the threshold. In this paper, simulations are carried out in the ideal case and under the assumption of stabilizer measurements, and in order to further explore the decoding effect, it is necessary to consider simulation experiments of the decoder of this paper under the depolarizing noise model, with the correlation between P_{physical} and P_{logical},

$$\rho \to (1-p)\rho + p_x X\rho X + p_y Y\rho Y + p_z Z\rho Z, \tag{15}$$

where p_x, p_y, and p_z are the probabilities of x, y, and z errors occurring on a qubit, respectively, $(1-p)$ is the probability that no error occurs. In this case, x, y, and z errors occur with equal probability, $p_x = p_y = p_z = 3/p$.

The measurements of the stabilizers are just what we call error syndromes. The main task of error correction is to determine the error correction operator applicable to a particular syndrome. This decoder, after analyzing all possible errors with respect to the probability of the observed syndrome, gives an error model and outputs a correction operator. The decoder takes the error syndrome as input and obtains the corresponding threshold based on the relationship between P_{physical}, P_{logical}, and code distance. The logical error rate is plotted within the range of physical error rates for different code distances, and the threshold is generally defined as the physical error rate value at their intersection. Given different code distances, corresponding thresholds can be obtained. For the code distance d, the logical error rate P_{logical} reads

$$P_{\text{logical}} = (p - P_{\text{threshod}}) \times d^{1/v_0}, \tag{16}$$

where P_{threshod} is the threshold of decoders, and P_{logical} is the logical error rate, and v_0 is the scaling exponent corresponding to the universality class of the three-dimensional random plaquette gauge model. When using the same dataset size, there is a constant execution time and physical error rate P_{logical} for a given code spacing, and the sum of samples from multiple datasets with different P_{physical} increases linearly relative to the code spacing. Training improves decoding performance because the samples are more correlated.

Yhe XZ^2 was decoded using a decoder in this study, with the logical failure rate as the physical error rate for different code distances d of the XYZ^2 code. For the accuracy range of the decoder in this paper, XYZ^2 gives an error threshold of about 19% or so, and is found to be P_{logical} as a function of the number of data qubits above the threshold, while for low error rates, the logical failure rate is instead proportional to the code distance.[50] As shown in Fig. 7, we give the logical failure rate and the physical error rate as a function of physical error rate for XYZ^2 codes in the presence of depolarizing noise, considering between $d = 3$ to $d = 5$ and $d = 7$ to $d = 11$, respectively. Comparing the physical error rate p as a function of the logical error rate P_{f} for XYZ^2 codes at these two code spacings respectively, we observe that for these two code spacings, the logical failure rate under depolarizing noise on XYZ^2 codes is shown in Fig. 7, as a function of the physical error rate. We can see that different code spacings produce different logical error rates. The longer the code pitch, the higher the success rate of error correction. In terms of threshold when the physical error rate is below the threshold, the larger the code distance, the greater the error rate can be corrected, and therefore the lower the logical error rate. While a larger code distance can correct more errors, the logical error rate will be greater and the code itself will require more quantum bits which will result in more errors. Although the decoding threshold has been improved to some extent, the threshold strength still needs to be further improved.

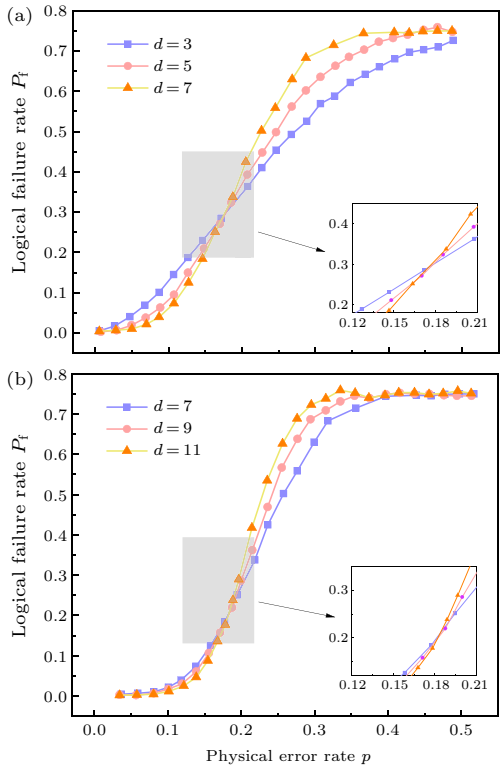


Fig. 7. (a) Correspondence as a function of physical error rate error, logical error rate, and code spacing $d = 3,5,7$. $P_{\text{threshold}} = 0.18856$. (b) Functional correspondence of physical error rate, logical error rate, and code distance $d = 7,9,11$. $P_{\text{threshold}} = 0.19043$. The horizontal coordinate represents the physical error rate and the vertical coordinate represents the logical error rate. For better numerical analysis, the different coding distances d in the two groups are marked in purple, pink, and orange, respectively.

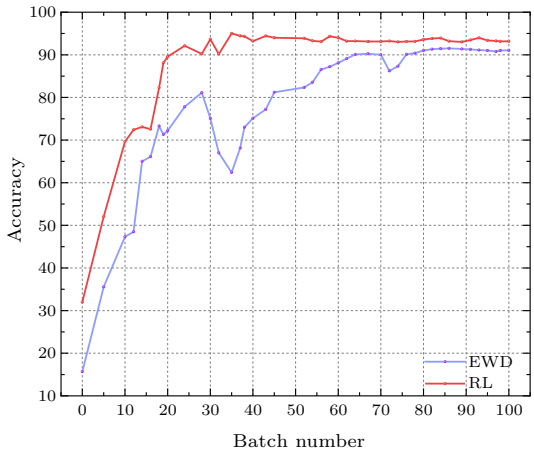


Fig. 8. We will group quantum bits into batches to compare the performance of the EWD decoder with the reinforcement learning model of the state-attention mechanism proposed in this study. The horizontal axis represents the data in each batch and the vertical axis represents the data accuracy.

Then in order to improve the effectiveness of error correction, qubits are usually grouped into batches and the model performance is evaluated for each batch of data. We consider comparing our reinforcement learning model containing the attention mechanism with the EWD decoder, and then evaluate the performance of the model proposed in this paper based on the accuracy of each batch of data. The results shown in Fig. 8 indicate that the decoder proposed in this paper performs stably and better than the EWD decoder.[30]

By selecting a set of suitable training datasets at different code spacings, compared with general reinforcement learning models, respectively, and configuring the parameters according to the experimental requirements. Error correction is performed using the trained network model in this paper. Performance evaluation is performed on different test datasets. From the threshold results and the performance analysis results of the decoder used in this paper, it is found that our reinforcement learning decoder, thanks to the attention mechanism, further improves the decoding performance. Then to quantify the performance of our decoder against the bias noise, the finite deviation noise under $\eta = 10$ will be explored again, which corresponds to a major error channel, and we consider the probability of error for any type p. In Fig. 9, we show the success rate of the decoder on the bias noise. We find the highest success rate for depolarizing noise, which is just what our decoder was trained for.

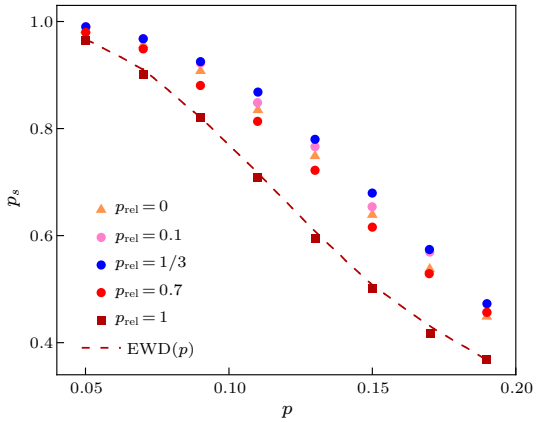


Fig. 9. Using a decoder trained with depolarizing noise ($p_{\text{rel}} = 1/3$), bias noise ($d = 5$), and a phase-flip error probability $p_z = p_{\text{rel}}p$, so $p_x = p_y = (1 - p_{\text{rel}})p/2$. For $p_{\text{rel}} = 1$, the corrector contains only the z error and we compare the decoder to EWD.

Finally, Table 1 lists the training time required for the various lattice sizes considered. All training was performed using a desktop computer.

Table 1. Training time for each reinforcement learning model.

Code distance	$d=3$	$d=5$	$d=7$	$d=9$	$d=11$
Training time (h)	0.33	5	12	32	70

6. Conclusion

This paper investigates a novel error correction method for decoding topological QEC codes based on XYZ^2 codes by reinforcement learning algorithms in the context of utilizing an attention mechanism. This mechanism is combined with reinforcement learning, where the threshold of the error correction code is also raised to about 0.19 at larger code spacing, and which enables reinforcement learning to focus the intelligence on the valuable states in the input state sequence during decision making, for example. The attended features, states,

will make the intelligent body learn faster and will learn a better strategy, which is an advantage over the EWD decoder. In addition, there are many shortcomings in this work, for example, the XYZ^2 code we studied. Reinforcement learning decoding can only be input in the form of squares, etc. Although the decoding efficiency of the reinforcement learning model with the addition of the attention mechanism in this paper is more improved, the improvement is small and does not reach the optimal situation. The quantum computing field to join the attention mechanism of the follow-up work still needs further research.

Acknowledgments

This work was supported by the Natural Science Foundation of Shandong Province, China (Grant No. ZR2021MF049) and Joint Fund of Natural Science Foundation of Shandong Province (Grant Nos. ZR2022LLZ012 and ZR2021LLZ001).

References

[1] He Z, Fan X, Chu P and Ma H 2020 *Acta Phys. Sin.* **69** 160301 (in Chinese)
[2] Knill E and Laflamme R 1997 *Phys. Rev. A* **55** 900
[3] Looi S Y, Yu L, Gheorghiu, V and Griffiths R B 2008 *Phys. Rev. A* **78** 042303
[4] Beale S J, Wallman J J, Gutiérrez M, Brown K R and Laflamme R 2018 *Phys. Rev. Lett.* **21** 190501
[5] Farhi E, Goldstone J and Gutmann S 2014 arXiv:1411.4028 [quant-ph]
[6] Li S, Long G, Bai F, Song L and Zheng H 2001 *Proc. Natl. Acad. Sci. USA* **98** 11847
[7] Varona S and Martin-Delgado M A 2020 *Phys. Rev. A* **102** 032411
[8] Kitaev A 1997 *Russ. Math. Surveys* **52** 1191
[9] Freedman M H and Meyer D A 2001 *Found. Comput. Math.* **1** 325
[10] Hu L, Wu S H, Cai W, *et al.* 2019 *Sci. Adv.* **5** eaav2761
[11] Kitaev A Y 2003 *Ann. Phys.* **303** 2
[12] Buerschaper O, Morampudi S C and Pollmann F 2014 *Phys. Rev. B* **90** 195148
[13] Kitaev A 2006 *Ann. Phys.* **321** 2
[14] Hermanns M, Kimchi I and Knolle J 2018 *Annu. Rev. Conden. Matter Phys.* **9** 17
[15] Fuentes P, Martinez J E, Crespo P M and Garciafrias J 2020 *Phys. Rev. A* **102** 012423
[16] Xue Y J, Wang H W, Tian Y B, Wang Y N, Wang Y X and Wang S M 2022 *Quantum Engineering* **2022**
[17] Fujisaki J, Oshima H, Sato S and Fujii K 2021 *Phys. Rev. Res.* **4** 043086
[18] Wang H, Song Z, Wang Y, Tian Y and Ma H 2020 *Quantum Inf. Process.* **21** 280
[19] Gao X and Duan L M 2017 *Nat. Commun.* **8** 662
[20] Carrasquilla J and Melko R G 2017 *Nat. Phys.* **13** 431
[21] Liu N and Rebentrost P 2018 *Phys. Rev. A* **97** 042315
[22] Davaasuren A, Suzuki Y, Fujii K and Koashi M 2020 *Phys. Rev. Res.* **2** 033399
[23] Andreasson p, Johansson J, Liljestrand S and Granath M 2019 *Quantum* **3** 183
[24] Bolens A and Heyl M 2021 *Phys. Rev. Lett.* **127** 110502
[25] Mills K, Spanner M and Tamblyn I 2017 *Phys. Rev. A* **96** 042113
[26] Zhang Y H, Zheng P L, Zhang Y and Deng D L 2020 *Phys. Rev. Lett.* **125** 170501
[27] Wang H W, Xue Y J, Ma Y L, Hua N and Ma H Y 2021 *Chin. Phys. B* **31** 010303
[28] Wootton J R and Loss D 2012 *Phys. Rev. Lett.* **109** 160503
[29] Hutter A, Wootton J R and Loss D 2014 *Phys. Rev. A* **89** 022326
[30] Hammar K, Orekhov A, Hybelius P W, Wisakanto A K, Srivastava B, Kockum A F and Granath M 2022 *Phys. Rev. A* **105** 042616
[31] Denil M, Bazzani L, Larochelle H, *et al.* 2012 *Neural Comput.* **24** 2151
[32] Waldherr G, Wang Y, Zaiser S, *et al.* 2014 *Nature* **506** 204
[33] Lee Y C, Brell C G and Flammia S T 2017 *J. Stat. Mech.* **2017** 083106
[34] Wootton J R 2015 *J. Phys. A: Math. Theor.* **48** 215302
[35] Srivastava B, Kockum A F and Granath M 2022 *Quantum* **6** 698
[36] Bahdanau D, Cho K and Bengio Y 2015 arXiv:1409.0473 [cs.CL]
[37] Soydaner D 2022 *Neural Comput. Appl.* **34** 13371
[38] Sethi K, and Madhav Y V, Kumar R and Bera P 2021 *J. Inf. Secur. Appl.* **61** 102923
[39] Lin T X, Su Z, Xu Q C, Xing R and Fang D F 2020 *IEEE Access* **8** 69284
[40] Andreasson P, Johansson J, Liljestrand S and Granath M 2019 *Quantum* **3** 183
[41] Cheung W C, Simchi-Levi D, Zhu R 2020 *Proceedings of the 37th International Conference on Machine Learning* **119** 1843
[42] Denil M, Bazzani L, Larochelle H and De F N 2012 *Neural Comput.* **24** 2151
[43] Nautrup H P, Delfpsse N, Dunjko V, Briegel H J and Friis N 2019 *Quantum* **3** 215
[44] Faist P, Nezami S, Albert V V, Salton G, Pastawski F, Hayden P, *et al.* 2020 *Phys. Rev. X* **10** 041018
[45] Varona S and Martin-Delgado M A 2020 *Phys. Rev. A* **102** 032411
[46] Kumar N, Swain S N and Murthy C S R 2018 *IEEE Trans. Network Service Manag.* **15** 718
[47] Roderick M, MacGlanshan J and Tellex S 2017 arXiv:1711.07478 [cs.LG]
[48] Mnih V, Kavukcuoglu K, Silver D, *et al.* 2015 *Nature* **518** 529
[49] Williams R J 1992 *Mach. Learn.* **8** 229
[50] Stephens A M 2014 *Phys. Rev. A* **89** 022321