New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG IOP Institute of Physics

Enhancing variational quantum state diagonalization using reinforcement learning techniques

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics

Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

CrossMark

OPEN ACCESS

RECEIVED
24 July 2023

REVISED
16 December 2023

ACCEPTED FOR PUBLICATION
5 January 2024

PUBLISHED
19 January 2024

PAPER

Enhancing variational quantum state diagonalization using reinforcement learning techniques

Akash Kundu[1,2,*], Przemysław Bedełek[3], Mateusz Ostaszewski[3], Onur Danaci[4,5], Yash J Patel[6], Vedran Dunjko[4,6] and Jarosław A Miszczak[1]

[1] Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Gliwice, Poland
[2] Joint Doctoral School, Silesian University of Technology, Gliwice, Poland
[3] Warsaw University of Technology, Institute of Computer Science, Warsaw, Poland
[4] Applied Quantum Algorithms, Lorentz Institute, Leiden University, Leiden, The Netherlands
[5] Leiden Institute of Physics, Leiden University, Leiden, The Netherlands
[6] Leiden Institute of Advanced Computer Science, Leiden University, Leiden, The Netherlands
[*] Author to whom any correspondence should be addressed.

E-mail: akundu@iitis.pl

Keywords: quantum state diagonalization, reinforcement learning, quantum architecture search, RL-VQSD, random search, entanglement spectroscopy

Abstract

The variational quantum algorithms are crucial for the application of NISQ computers. Such algorithms require short quantum circuits, which are more amenable to implementation on near-term hardware, and many such methods have been developed. One of particular interest is the so-called variational quantum state diagonalization method, which constitutes an important algorithmic subroutine and can be used directly to work with data encoded in quantum states. In particular, it can be applied to discern the features of quantum states, such as entanglement properties of a system, or in quantum machine learning algorithms. In this work, we tackle the problem of designing a very shallow quantum circuit, required in the quantum state diagonalization task, by utilizing reinforcement learning (RL). We use a novel encoding method for the RL-state, a dense reward function, and an ϵ-greedy policy to achieve this. We demonstrate that the circuits proposed by the RL methods are shallower than the standard variational quantum state diagonalization algorithm and thus can be used in situations where hardware capabilities limit the depth of quantum circuits. The methods we propose in the paper can be readily adapted to address a wide range of variational quantum algorithms.

1. Introduction

In the last few decades, researchers from various scientific disciplines have come together to study and develop quantum algorithms and their experimental realization. Among the originally proposed quantum algorithms, many require millions of physical qubits to be implemented on quantum hardware to deal with instance sizes of real-world importance. Unfortunately, the existing quantum hardware is limited to the order of a few hundred physical qubits, and these are called noisy intermediate-scale quantum (NISQ) devices. The NISQ algorithms are small and prone to noise and decoherence, and thus, one needs to consider Variational Quantum Algorithms (VQAs) that can work under such restrictions.

Among the class of VQAs, variational quantum state diagonalization (VQSD) [1] is an algorithm that utilizes a quantum–classical hybrid procedure to identify the unitary rotation under which a given quantum state becomes diagonal in the computational basis, i.e. it diagonalizes a quantum state. It has several applications, including quantum state fidelity estimation [2], device certification [3], Hamiltonian diagonalization [4], and as a method to extract entanglement properties of a system [1, 5]. VQSD generalizes the well-studied problem of quantum state preparation, which can be understood as quantum state tomography for pure states[7]. Considering it has applications that range from quantum information to condensed matter physics, an efficient way to deal with quantum state diagonalization may lead to interesting insights in these fields.

We note that there exist algorithmic exact methods for quantum state diagonalization based on quantum principal component analysis (qPCA) [6]. However, they lead to deeper circuits that could, in principle, be obtained with variational methods. However, to achieve this, the most challenging aspect of VQSD is to construct an *efficient* ansatz (which refers to the unitary) that diagonalizes a given quantum state. For the analysis in this paper, we consider the following factors as the indicators for *ansatz efficiency*: (1) the depth, understood as the number of parallel operations in the ansatz; (2) the total number of quantum gates, and (3) the accuracy in the estimation of eigenvalues.

In the standard VQSD methods [1, 5], a layered hardware efficient ansatz (LHEA) is utilized. A single layer of the ansatz contains two-qubit gates acting on neighbouring qubits. Although the LHEA parameter count increases linearly with the number of layers and qubits, it has trainability issues and often encounters local minima [1]. To tackle the trainability issue, instead of using a fixed structure of LHEA, the authors allow additional updates (i.e. changes in the ansatz structure) during the classical optimization process [1]. In this process, every optimization step minimizes the cost function with a small random change to the ansatz structure. The new structure is approved or rejected based on a simulated annealing scheme [7]. Although the varying structure LHEA outperforms fixed structure LHEA, the number of gates in the quantum circuit increases rapidly as we scale the size of the quantum state. Hence, the problem of finding a method to construct an ansatz that satisfies all efficiency criteria is still an open problem.

In the case of some VQAs, to address the challenges of finding the architecture of ansatz, methods have been introduced that draw on the insight and techniques of machine learning [8–12], such as a process of automating the architecture engineering of quantum circuits is known as quantum architecture search (QAS) [9, 13, 14]. Recent studies have strongly suggested that double deep Q-networks (DDQN) in reinforcement learning (RL) can successfully solve QAS problems [8, 10], performance improvement in QAOA variants [15] as well as the task of quantum compiling [16].

Contributions

Following the above line of work, we introduce a RL driven VQSD method (i.e. RL-VQSD), which automates the search for optimal succinct ansatz (i.e. RL-ansatz). The RL-VQSD algorithm constitutes:

1. A novel depth-based binary encoding scheme [17] to encode the RL-state.
2. A dense reward function, which we introduce in the paper crafted particularly for the task of quantum state diagonalization.
3. A DDQN with an ϵ-greedy policy for better stability.

Using these components we demonstrate that the ansatz proposed by the RL-agent can successfully diagonalize arbitrary mixed quantum states of full-rank with a smaller number of gates and depth compared to the existing ansatz structures. We exemplify the functioning of the RL-VQSD by diagonalizing the quantum states arising in condensed matter physics while maintaining a short depth and gate count of the resulting RL-ansatz. Moreover, a deeper investigation reveals that the combination of the binary encoding of the RL-state and the dense reward function is responsible for the success of diagonalizing larger quantum states. Finally, we demonstrate the hardness of the problems by utilizing a random agent in the VQSD algorithm and show the performance of the random agent significantly decreases as we scale up the qubits in the quantum state. Moreover, we show that the RL-agent not only provides us with a more consistent outcome, but it gives significantly better circuit depth, gate count, and approximation quality compared to the random agent.

The rest of this paper is organized as follows. In section 2, we review the standard methods for variational quantum state diagonalization and provide an overview of the ansatz construction, and reinforcement learning (RL). In section 3, we describe the proposed scheme for the construction of variational quantum state diagonalization circuits, including the method for encoding quantum circuits, the dense reward function, and the performance comparison of the encoding and the reward. Section 4 summarises the numerical results obtained to demonstrate the application of the proposed RL-VQSD. Finally, in section 5, we briefly summarize the contribution and provide some remarks concerning the possible extension of the introduced approach.

[7] If one is given a quantum state $|\psi\rangle$, then VQSD can potentially find a short-depth circuit that approximately prepares $|\psi\rangle$.

2. Preliminaries

This section briefly reviews the standard methods of variational quantum state diagonalization. We also outline the standard procedure of constructing an ansatz and introduce basic concepts from RL.

2.1. Variational quantum state diagonalization

Classical methods for diagonalization typically scale polynomially with the dimension of the matrix [18]. Similarly, the number of measurements required for quantum state tomography scales polynomially with the dimension of the Hilbert space. Moreover, as discussed, the qPCA is costly to implement in NISQ devices.

To tackle these issues, a hybrid quantum–classical method for quantum state diagonalization—VQSD – has been proposed in [1]. For a quantum state ρ, the algorithm is composed of three subroutines:

- TRAINING In this subroutine, for a given state ρ, one optimizes the parameters $\vec{\theta}$ of a quantum gate sequence $U(\vec{\theta})$, which (ideally) after optimization satisfies

$$\rho' = U\left(\vec{\theta}_{\text{opt}}\right)\rho U\left(\vec{\theta}_{\text{opt}}\right)^{\dagger} = \rho_{\text{diag}}, \tag{1}$$

  where ρ_{diag} is the diagonalized ρ in its eigenbasis and $\vec{\theta}_{\text{opt}}$ are the optimal angles. One can utilize classical gradient-based methods such as SPSA and Gradient-Descent or gradient-free optimization methods such as COBYLA [19] and POWELL [20] in the training process.
- EIGENVALUE READOUT In this subroutine, using the optimized unitary $U(\vec{\theta}_{\text{opt}})$ and one copy of state ρ, one can extract—for low-rank states—all the eigenvalues or—for full-rank state—the largest eigenvalues. This is achieved by measuring the ρ' in the computational basis, $\mathbf{b} = b_1 b_2 \ldots b_n$, as follows

$$\lambda' = \langle \mathbf{b}|\rho'|\mathbf{b}\rangle, \tag{2}$$

  where λ' are inferred eigenvalues.
- EIGENVECTOR PREPARATION In the final step, one can prepare the eigenvectors associated with the largest eigenvalues. If $\mathbf{b}'$ is a bit string associated with λ' then one can get the inferred eigenvectors $|v'_{\mathbf{b}'}\rangle$ as follows

$$|v'_{\mathbf{b}'}\rangle = U\left(\theta_{\text{opt}}\right)^{\dagger}|\mathbf{b}'\rangle = U\left(\theta_{\text{opt}}\right)^{\dagger}\left(X^{b_1}\otimes\ldots\otimes X^{b_n}\right)|0\rangle. \tag{3}$$

The workflow in the VQSD procedure is illustrated in figure 1.

The cost function proposed in [1] as a part of the training process is a function of the *purity* of the state that needs to be diagonalized. It takes the following form

$$C\left(\vec{\theta}\right) = \text{Tr}\left(\rho^2\right) - \text{Tr}\left(\mathcal{D}\left(\rho'\right)^2\right), \tag{4}$$

where $\mathcal{D}$ denotes a dephasing channel, that eliminates the off-diagonal elements. When $C(\vec{\theta})$ is sufficiently close to zero, one can say that the quantum state is diagonalized. It should be noted that there are many ways to define a cost function that quantifies how far ρ' is from being diagonal [21]. However, due to computational purposes, we choose the cost function of the form given in equation (4).

2.2. Ansatz construction

In the TRAINING subroutine of figure 1, the correct choice of the ansatz is crucial, as it is the main factor determining whether the diagonalization task can be performed. Additionally, the choice of the ansatz can also impact the execution of the EIGENVALUE READOUT and EIGENVECTOR PREPARATION, as one has to use it in both cases.

In many instances of VQAs, the structure of the ansatz is dictated by the underlying problem. For example, in Variational Quantum Eigensolver (VQE) [22] and the Quantum Approximate Optimization Algorithm (QAOA) [23], the ansatz can be defined based on the problem Hamiltonian. In VQE, the ansatz is constructed through the so-called Unitary Coupled Cluster (UCC) [24–26] method, and in QAOA, it is given by first-order Trotterization of the time-dependent Hamiltonian corresponding to the adiabatic preparation of the ground state. However, this is not the case for the VQSD algorithm; for an arbitrary unknown quantum state, the algorithm has no problem-inspired ansatz.

In the previous works [1, 2] to solve the optimization part, the authors proposed a fixed structure for an ansatz, namely LHEA. This type of ansatz is depicted in figure 2 where each layer $L \in [1..l]$ of $U_L(\vec{\theta})$ consists of a set of optimization parameters $\vec{\theta} \equiv \theta_i^j$, where i denotes the total number of layers and j is the number of parameters per layer. Each layer consists of two-qubit rotation gates which follow a periodic boundary

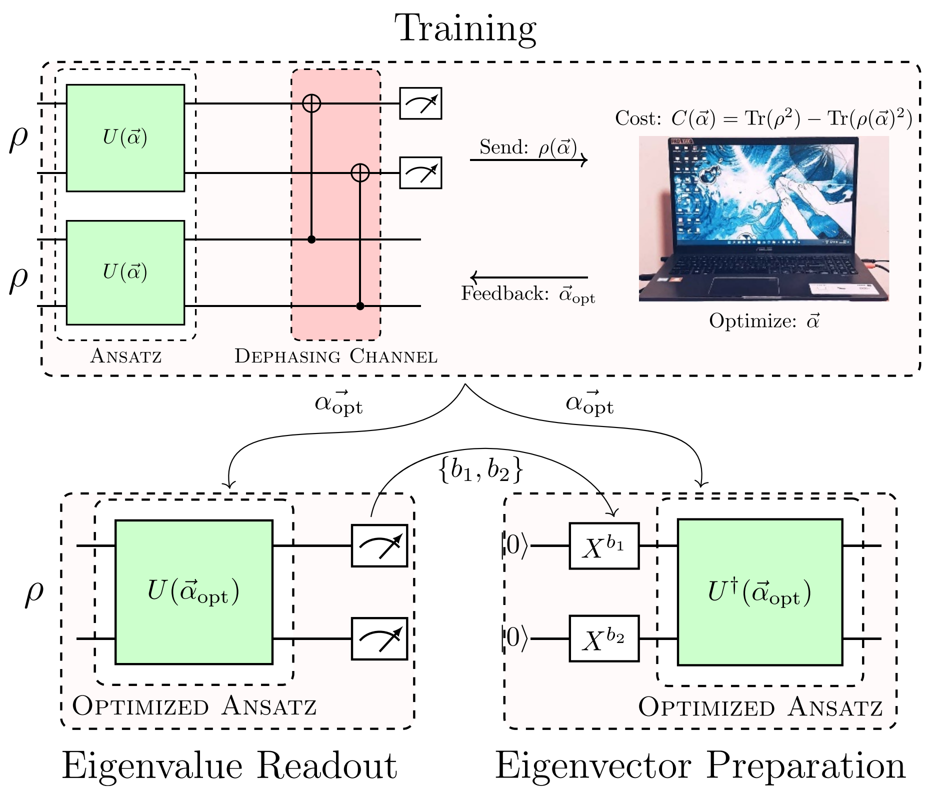

Training
ρ
ρ
$U(\vec{\alpha})$
$U(\vec{\alpha})$
Ansatz
Dephasing Channel
Send: $\rho(\vec{\alpha})$
Feedback: $\vec{\alpha}_{\text{opt}}$
Cost: $C(\vec{\alpha}) = \text{Tr}(\rho^2) - \text{Tr}(\rho(\vec{\alpha})^2)$
Optimize: $\vec{\alpha}$
$\vec{\alpha_{\text{opt}}}$
$\vec{\alpha_{\text{opt}}}$
$\{b_1, b_2\}$
ρ
$U(\vec{\alpha}_{\text{opt}})$
Optimized Ansatz
$|0\rangle$
$|0\rangle$
X^{b_1}
X^{b_2}
$U^\dagger(\vec{\alpha}_{\text{opt}})$
Optimized Ansatz
Eigenvalue Readout
Eigenvector Preparation

Figure 1. Elements of Variational Quantum State Diagonalization (VQSD) algorithm. In the presented example, we consider the diagonalization for the 2-qubit input state. It should be noted that to diagonalize the N qubit quantum state the algorithm requires $2N$ number of qubits in the algorithm.

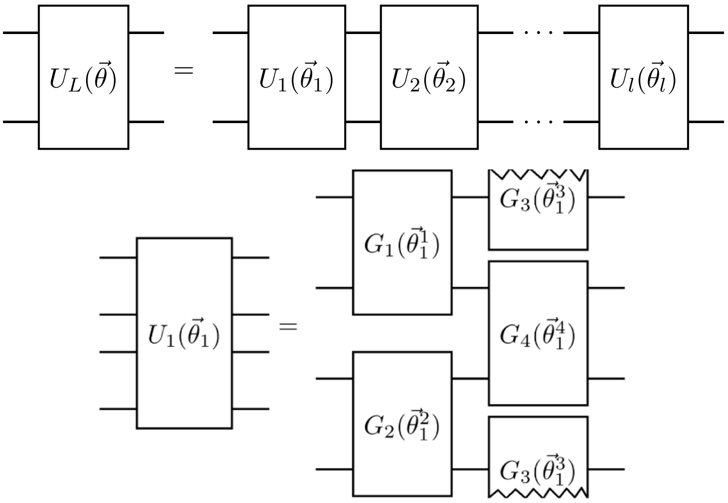


Figure 2. Structure of a layered hardware efficient ansatz, where the ansatz $U_l(\vec{\theta})$ is decomposed into layer-wise unitaries $U_l(\vec{\theta}_l)$ for $l = 1, 2, \ldots, l$. Each gate $U_l(\vec{\theta}_l)$ is further decomposed into two-qubit rotations. For $\vec{\theta}_i^j$, index i denotes the layer number, and j is the index specifying the parameter count.

condition. In LHEA, there are two possible ways to construct the two-qubit parameterized gates, which are depicted in figure 3.

Instead of diagonalizing with a fixed structure ansatz, one can allow it to vary during the optimization process. This scenario starts from a two-qubit parameterized gate on random qubits, and then the gate sequence is optimized by minimizing the cost function and changing the gate-set structure. Hence, the gate sequence is allowed to grow if the algorithm fails to minimize the cost function for a specified number of iterations. Then, one adds an identity gate spanned by new variational parameters that are randomly added to the ansatz. This step is equivalent to adding a layer to the ansatz. This method is discussed in more detail in [7].

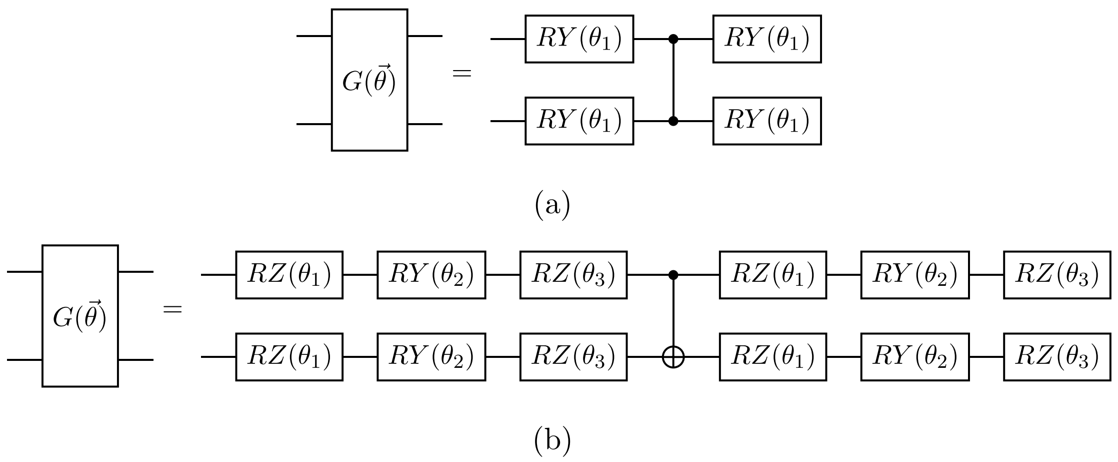


Figure 3. Two possible decompositions of the two-qubit rotations in each layer-wise unitary $U_i(\vec{\theta}_i)$. It can be constructed into two forms with (a) one and (b) three parameters respectively.

To address the lack of a definitive structure for diagonalizing unitary, in this paper we utilize RL to automate the exploration for an efficient ansatz construction.

2.3. Reinforcement learning

In reinforcement learning (RL), an agent interacts with its environment to learn an optimal policy by trial and error approach [27]. An RL process can be modeled as a Markov decision process defined by the tuple (S, A, P, R), where S and A represent the state and action spaces, the function $P : S \times S \times A \to [0,1]$ defines the transition dynamics, and $R : S \times A \to \mathbb{R}$ describes the reward function of the environment. In this work, we consider the action A and the state S to be finite and discrete sets. An episode describes all interactions between an agent and its environment until a user-specified termination condition is met.

An agent's behavior in the environment is governed by a stochastic policy $\pi(a|s) : S \times A \to [0,1]$, for $a \in A$ and $s \in S$. The metric that assesses an agent's performance is given by the *return* and takes the form of a discounted sum as follows

$$G(\tau) = \sum_{j=0}^{T-1} r_j \gamma^{j+1},$$

where $\tau = (s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1}) \in (S \times A \times \mathbb{R})^T$ is the interaction sequence, T is a fixed length called horizon, and γ is an environment-specific discount factor. The agent's objective is to determine the optimal policy that maximizes the expected return.

In a large unknown environment, the agent needs to be able to adapt to many different situations and develop multiple strategies at the same time. Hence, highly expressive function approximators such as deep neural networks to parametrize the agent's policy π can be advantageous.

Here, we settled on using a DDQN [28]. DDQN is a Q-learning algorithm based on the standard DQN [29], which features two neural networks (NN) to increase the stability of the prediction of Q-values for each state and action pair. We represent the state space as an ordered list of layers that are composed of a single depth of the quantum circuit. An action space is defined by a list of four numbers, corresponding to `RX`, `RY`, `RZ`, and `CNOT` quantum gates. For the sake of brevity, we defer the detailed description of the DDQN algorithm in appendix A.

2.4. Error quantification

To quantify the eigenvalue error throughout the paper, we use the following figure of merit [1]

$$\Delta_i = \sum_{i=1}^{m} (\lambda_i - \lambda_i')^2, \tag{5}$$

where m represents the number of the largest eigenvalues, λ_i is the true eigenvalue and λ_i' is the inferred eigenvalue obtained from the EIGENVALUE READOUT subroutine. In the ideal case, where the state is completely diagonalized, $m = 2^n$ indicates all the eigenvalues have been considered. Throughout the paper, we set $m = 2^n$ if not specified explicitly otherwise.

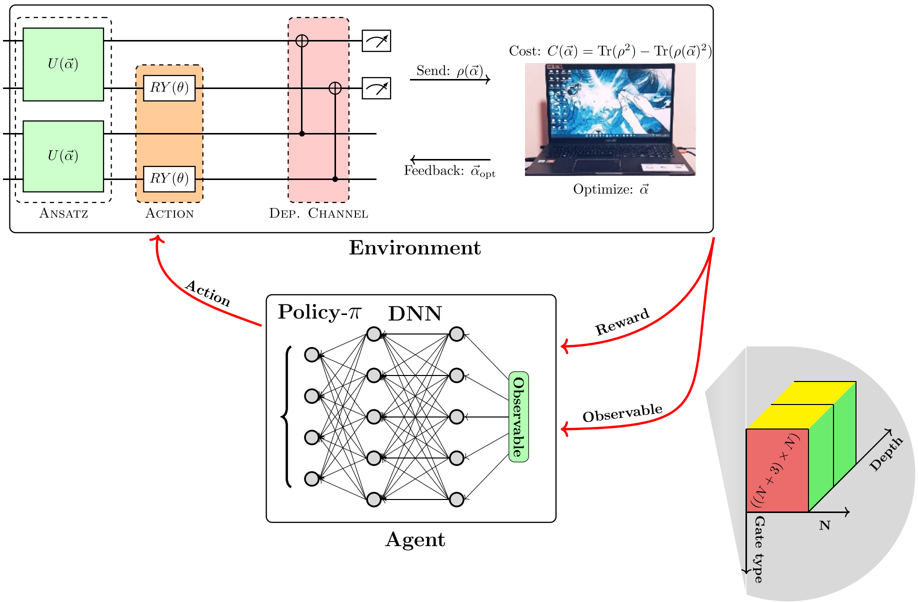


Figure 4. Illustration of the RL-VQSD process. In this process, the VQA is represented as the environment and the ansatz as the RL-state. The RL-agent receives the optimized cost function in the form of a reward and the RL-state from the environment. Following an ϵ-greedy policy, the agent then decides on an action (i.e. a quantum gate), which in the next step updates the RL-state. Utilizing the new RL-state the VQA optimizes the cost function and generates a new reward function to feed it to the agent. This process is repeated until all the steps in an episode are exhausted, or the cost function reaches a predefined threshold value. Throughout the paper, we start the RL-VQSD with an empty circuit and at each step, the agent chooses an action to construct the RL-ansatz, indicating $U(\vec{\alpha}) = \mathbb{I}$.

3. Proposed approach

In this section, we give the details of the proposed RL-VQSD as depicted in figure 4. In our modeling of the algorithm, the states of the environment encode the possible architectures of the quantum circuit (i.e. the ansatz), and the actions correspond to a gate. At first, we briefly discuss the sub-components of the RL-VQSD which include (1) a binary encoding for the RL-state, (2) a one-hot encoding to define actions, and (3) the engineering of a dense reward function. Next, we discuss the agent-environment settings and hyperparameters relevant to RL-VQSD. Finally, we benchmark the performance of the binary encoding scheme and the dense reward function in comparison with the encoding and reward proposed in [8], showing that the success of RL-agent is heavily dependent on a well-engineered encoding scheme for RL-state and reward function.

3.1. Encoding scheme for state

Motivated by the ideas in [8, 30], in [17] a binary encoding scheme was introduced. In this scheme, the gate structure of the ansatz is expressed as a tensor of dimension $[D_{\text{max}} \times ((N+3) \times N)]$, where N represents the size of the problem and D_{max} is the considered maximum depth of the ansatz. For VQSD, N represents the number of qubits in the quantum state that need to be diagonalized. The proposed encoding can be explained through the following two points:

1. **Freedom in connectivity** The encoding enables *all-to-all* qubit connectivity, but it can be restricted by considering *unidirectional nearest neighbour* connections only. In this scenario, the matrix dimension $((N+3) \times N)$ is reduced to $(4 \times N)$. One should note that in the case of a two-qubit gate, one is not required to keep track of the control and target simultaneously. Hence, defining one argument of the two-qubit gate implicitly provides information about the other argument due to its nearest neighbour and unidirectional nature. A similar encoding scheme is described in [30].
2. **Depth-based encoding** In previous work [8] each $((N+3) \times N)$ matrix carries information corresponding to each action taken by the agent, where each action represents either a single or a two-qubit gate. Additionally, the information was integer-based, in the range 0 to N.
   On the contrary, In our work, the encoding is binary and depth-based. For example, if $D_{\text{max}} = 3$, then the encoding initiates by filling up the $[i \times ((N+3) \times N)]$ for $i = 1$ until a depth of RL-ansatz is encoded.

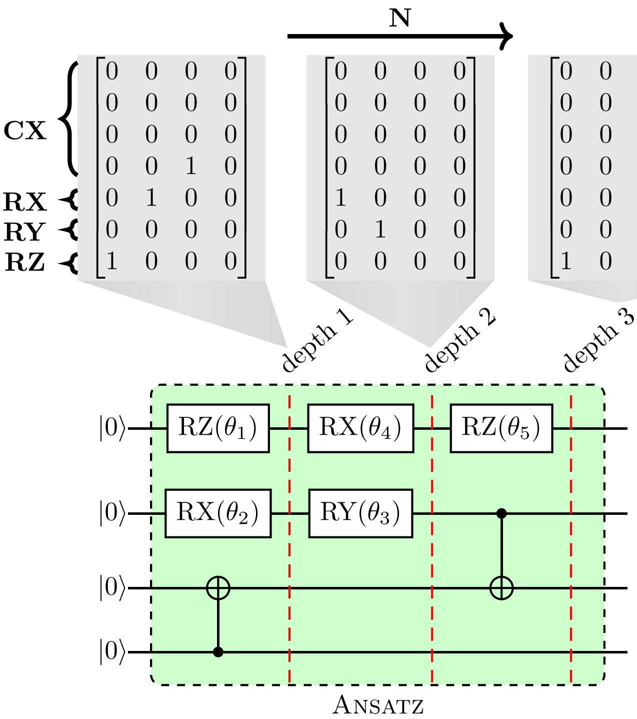


Figure 5. Example of the proposed encoding for a 4-qubit ansatz into an RL-state. The first $(N\times N)$ square matrix is reserved for the CNOT connectivity. The columns of the square matrix encode the *target qubit*, and the rows represent *control qubits*. The remaining $((N+j)\times N)$ elements encode arbitrary rotation towards j direction where $j=1$, 2, and 3, for X, Y and Z rotations, respectively.

Thus, we have

$$\left[\underbrace{((N+3)\times N)}_{\text{depth}=1},\underbrace{((N+3)\times N)}_{\text{all zeros}},\underbrace{((N+3)\times N)}_{\text{all zeros}}\right]. \tag{6}$$

Then, as $i=1$ is filled up, we move to $i=2$ to encode depth $=2$ of the RL-ansatz, which yields

$$\left[\underbrace{((N+3)\times N)}_{\text{depth}=1},\underbrace{((N+3)\times N)}_{\text{depth}=2},\underbrace{((N+3)\times N)}_{\text{all zeros}}\right]. \tag{7}$$

Finally, the depth $=3$ is encoded in $i=3$ resulting in

$$\left[\underbrace{((N+3)\times N)}_{\text{depth}=1},\underbrace{((N+3)\times N)}_{\text{depth}=2},\underbrace{((N+3)\times N)}_{\text{depth}=3}\right]. \tag{8}$$

Each depth encoding follows the scheme shown in figure 5.

3.2. Actions

For constructing the quantum circuits, we use the scheme developed in [8] with CNOT and one-qubit rotation gates, which are feasible on currently available quantum devices. The encoding of the action space can be defined as follows. The CNOT gates are represented by a pair of values that indicate the positions of the control and target qubits, with enumeration starting from 0. As for the rotation gates, they are encoded using two integers, also starting from 0. The first integer identifies the qubit register, while the second integer specifies the rotation axis. For an N-size quantum state, the agent can choose from $3\times N$ single-qubit gates and $2\times\binom{N}{2}$ two-qubit gates. As we are utilizing deep RL methods, we employ the one-hot encoding technique to represent actions within the action space. Mathematically, one-hot encoding can be identified as the Kronecker–Delta function as follows [31]. Suppose x is a discrete categorical random variable that takes n distinct values $x_1,\ldots,x_n$. Then the one-hot encoding of a particular value x_i is a vector v where every component of v is zero except for the ith component, which has the value 1. We refer the reader to figure 5 for a visual illustration of one-hot encoding for the actions (shaded in grey color).

3.3. Reward function

To guide the agent quickly towards the goal, we introduce a reward that is dense in time at each time step t. The reward used in this work is given as

$$R = \begin{cases} +\mathcal{R} & \text{for } C_t\left(\vec{\theta}\right) < \zeta + 10^{-5} \\ -\log\left(C_t\left(\vec{\theta}\right) - \zeta\right) & \text{for } C_t\left(\vec{\theta}\right) > \zeta \end{cases}, \tag{9}$$

where the goal of the agent is to reach the minimum error for a predefined threshold ζ, i.e. the tolerance for cost function minimization. The ζ is a hyperparameter of the model. The cost function at each step t is calculated for the ansatz which outputs a state $\rho_t(\vec{\theta})$ as

$$C_t\left(\vec{\theta}\right) = \mathrm{Tr}\left(\rho^2\right) - \mathrm{Tr}\left(\rho_t\left(\vec{\theta}\right)^2\right). \tag{10}$$

3.4. Agent and environment specification

In this work, we use a [32] DDQN for better stability with an ϵ-greedy policy and the ADAM optimizer [33] to optimize the weights of the NN. More details about the RL procedure are described in the next section. As mentioned in the previous section, to obtain a reward R for the circuit (i.e. for each environmental state), an optimization subroutine needs to be applied to determine the values of the rotation gate angles. We use well-developed methods for continuous optimization, such as Constrained Optimization By Linear Approximation [19] (COBYLA), which we utilize to optimize the parameters of the quantum circuit.

4. Numerical demonstrations

4.1. Setup details

We start with the parameter specifications given in [8], which uses the DDQN algorithm with a discount factor of $\gamma = 0.88$ and an ϵ-greedy policy for selecting random actions. The value of ϵ is gradually decreased from 1 to a minimum value of 0.05 by a factor of 0.99995 at each step. The size of the memory replay buffer is set to 2×10^4, and the target network in the DDQN training is updated with every 500 action. Following each training episode, we conduct a testing phase where the probability of selecting a random action is set to 0, and the experience replay procedure is turned off. Experiences obtained during the testing phase are not added to the memory replay buffer. The source code and the specifications of the numerical experiments presented in this section are available from the publicly accessible code repository [34].

4.2. Experiment details

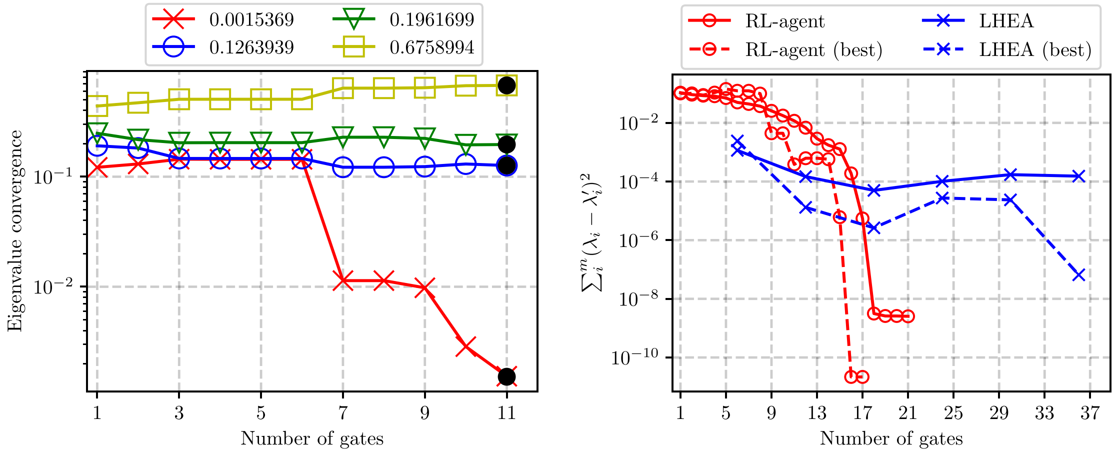
In the following numerical simulations, we consider 2-qubit random quantum states and the reduced ground state of a 3-qubit Heisenberg model to benchmark the performance of RL-VQSD in comparison with the VQSD method with l layers of LHEA. Further, to show the scaling of RL-VQSD we consider diagonalizing the reduced ground state of the 4-qubit Heisenberg model. For the experiment, we consider 10 000 episodes (if not stated otherwise) where each episode is decomposed into N_s steps. The value of N_s is set to 20, 40 and 60 while diagonalizing 2, 3 and 4-qubit problems respectively. In each step of an episode, the RL-agent decides on an action following the encoding provided in section 3.2, and then the action is translated into either rotation or `CNOT` gate. The value of the parameter for a new rotation gate is always initialized with 0. Then, the parameters of the circuit are optimized. In the next step, when the RL-agent decides on a new action, the new rotation is initialized to 0, but the previous rotation gates are set to their optimized angles. Then, the modified ansatz is optimized in the classical optimization subroutine (using COBYLA optimizer). This process is repeated until the problem is solved or all the steps in an episode are exhausted. Mainly for this work, at each step of an episode, we optimize all angles at once (global strategy), which we call global COBYLA. In diagonalizing 2-, 3- and 4-qubit quantum state we use 400, 500 and 1000 iterations of COBYLA optimizer respectively.

4.3. Analysis of RL-state encoding and reward function

Before diving into a rigorous investigation of the performance of RL-VQSD, we showcase the effectiveness of the RL-state encoding method (provided in section 3.1) along with the dense reward function (as in equation (9)). To benchmark the effectiveness, we compare the RL-ansatz proposed by the agent utilizing the following two settings to diagonalize 2- and 3-qubit quantum state with RL-VQSD: (1) the binary encoding

Table 1. Comparison of the binary encoding along with the reward function presented in the paper with the integer-based encoding and the sparse reward function presented in [8]. We diagonalize a 2-qubit arbitrary state and the reduced ground state of the 3-qubit Heisenberg model. The result shows that for 2-qubit, both the settings perform equivalently, but as we scale up the system to 3-qubit, the integer encoding with sparse reward fails to give an efficient RL-ansatz with small gates and depth that can help us achieve a good approximation of the eigenvalues. This study leads us to conclude that the success of an RL-agent significantly depends on appropriately encoding the RL-state and designing the reward function.

Settings	Qubits	Avg. err.	Avg. 1q gate	Avg. 2q gate	Avg. depth
Binary encoding section 3.1 and dense reward (9)	2	5.23×10^{-6}	14.40	4.60	14.12
	3	9.16×10^{-5}	18.19	12.08	20.32
Integer encoding and sparse reward [8]	2	6.51×10^{-6}	13.96	4.72	14.81
	3	5.97×10^{-4}	15.49	24.50	35.65



(a) Single mixed quantum state.

(b) Average over 50 random quantum state.

Figure 6. The summary of results for diagonalizing full rank 2-qubit random density matrix. In (a) we illustrate eigenvalue convergence for the diagonalization of a single mixed quantum state. In (b) we compare the performance of the RL-agent-generated ansatz with the LHEA. It can be seen that the RL-agent-generated ansatz gives us a better approximation of the eigenvalues. Additionally, the RL-based methods can achieve the accuracy of the LHEA using the circuit with significantly reduced depth of the resulting circuit.

scheme along with the dense reward function presented in this paper and (2) the integer-based RL-state encoding scheme (which we call *integer encoding*) and the sparse reward function described in [8] in solving quantum chemistry problems. In both cases, we keep the agent and environment specifications unchanged. Through this investigation, we show how the RL-state encoding and the engineering of the reward function are responsible for the success of the RL-VQSD method. In table 1, we present the results, which confirm that the integer encoding, along with sparse reward in [8], underperforms in finding a more accurate diagonalization of the state with a smaller number of gates and depth as the size of the diagonalizing state increases. Furthermore, it does not solve the diagonalization problem with a 10^{-4} threshold, a problem easily tackled by binary encoding and the dense reward methods.

4.4. 2-qubit random quantum states

In the first numerical experiment, we utilize RL-VQSD to diagonalize (1) a single mixed quantum state and (2) 50 random quantum states of the full rank of 2-qubit, to get the average eigenvalue approximation error and count the gates in RL-ansatz. We utilized the `random_density_matrix` of the module `quantum_info` of `qiskit` [35] to sample the quantum states from the Haar measure. By (1), we argue that RL-VQSD can exactly diagonalize a quantum state. The results of (2) demonstrate that the average performance of RL-VQSD is better than state-of-the-art ansatz.

In figure 6(a) we show that the agent can propose an ansatz that provides us with the exact eigenvalues for a 2-qubit random quantum state with 12 gates, containing 10 rotations and 2 `CNOT` gates. The RL-ansatz is depicted in figure 7.

Meanwhile, in figure 6(b), we benchmark the performance of RL-ansatz against LHEA. In the illustration, we show that the agent not only gives us a small ansatz to diagonalize with a specific predefined threshold $\zeta = 10^{-5}$ but also helps us achieve a lower error in eigenvalue estimation compared to LHEA. Furthermore, in table 2, we provide a rigorous comparison of the RL-ansatz and 6 layers of LHEA (of the

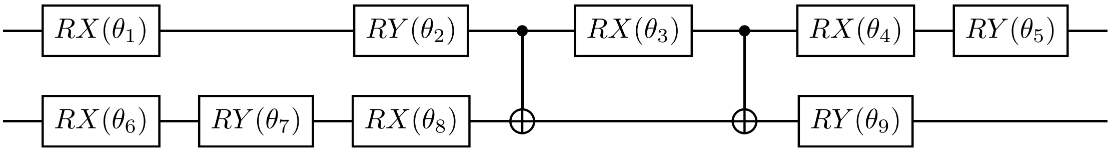


Figure 7. The ansatz proposed by RL-agent to diagonalize the 2-qubit state with eigenvalues convergence illustrated in figure 6(a). This shows us that even with very few gates and small depth the RL-VQSD can give us accurate diagonalization of small quantum systems.

Table 2. Comparison of the RL-ansatz (proposed by the RL-agent) and 6 layers of LHEA (utilized in the VQSD) used for diagonalizing 2-qubit arbitrary quantum states. For the comparison, we consider investigating the average error in eigenvalue estimation (Avg. error), the average number of single-qubit gates (Avg. 1q gate), the average number of the two-qubit gate (Avg. 2q gate), the average depth (Avg. depth) and the average number of total gates (Avg. total gate). The structure of the LHEA (depicted in figure 3(b)) utilized for the investigation is of a fixed structure whose depth and gates scales as $l\times$ (depth or number of gates) where l denotes the layers of the LHEA. It should be noted that the average is taken over 50 random quantum states. N.A. denotes not applicable.

	Layer	Avg. error	Avg. 1q gate	Avg. 2q gate	Avg. depth	Avg. total gate
RL-ansatz	N.A.	9.33×10^{-7}	10.68	2.28	9.54	14.06
6 layers of LHEA	1	2.42×10^{-4}	12	1	7	13
	2	1.31×10^{-5}	24	2	14	26
	3	4.98×10^{-5}	36	3	21	39
	4	1.02×10^{-4}	48	4	28	52
	5	1.72×10^{-4}	60	5	35	65
	6	1.53×10^{-4}	72	6	42	78

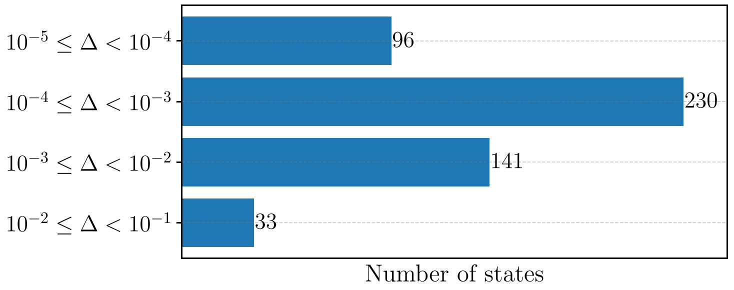


Figure 8. Statistics of error in eigenvalue estimation for 500 arbitrary quantum states. As an ansatz to diagonalize all the 500 random quantum states, we consider a fixed structure provided by the RL-agent. In this case, we consider the structure given in figure 7.

structure depicted in figure 3(b)). For the comparison, we evaluate the average statistical error (which is defined in equation (5)) in estimating eigenvalues, the mean count of one and two-qubit gates, the average depth, and the overall average count of gates as metrics. From the table, we can conclude that the average error of LHEA gets stuck around 10^{-5}, where the 2nd and the 3rd layers of LHEA provide the lowest error in eigenvalue estimation. Meanwhile, the RL-ansatz can, on average, give 10^2 times less error compared to LHEA with an ansatz composed of 3 times fewer parameterized gates and smaller depth. In table 2, for depth 14 LHEA (which corresponds to the 2nd layer of LHEA with 26 gates), we see that the average error in eigenvalue reaches to 1.31×10^{-5}. On the other hand, for the same depth, the RL-ansatz can achieve an error of 9.33×10^{-7} with an average (over the 50 random states) of 2.56 2-qubit and 11.58 1-qubit gates.

Furthermore, we explore the possibility of utilizing the ansatz proposed by the RL-agent, trained on a specific quantum state, to diagonalize random quantum states that differ from the initial fixed state. We can confirm that this is indeed possible in the case of the 2-qubit state. The corresponding results are presented in figure 8. One can argue that, in this case, the diagonalization task is relatively easy. However, our results show that it is possible to harness the RL-ansatz for a particular quantum state to diagonalize an arbitrary state of the same dimension. To conduct this experiment, we start by selecting a random quantum state and training it using the RL-agent. The RL-agent then provides us with an RL-ansatz specifically designed for that particular state. By utilizing this RL-ansatz as a quantum circuit and employing VQSD (refer to figure 1), we successfully diagonalize 500 arbitrary quantum states. Our results indicate that the RL-ansatz achieves a reasonable accuracy, with the majority of quantum states falling within the range of $10^{-4}\leqslant\Delta\leqslant10^{-3}$.

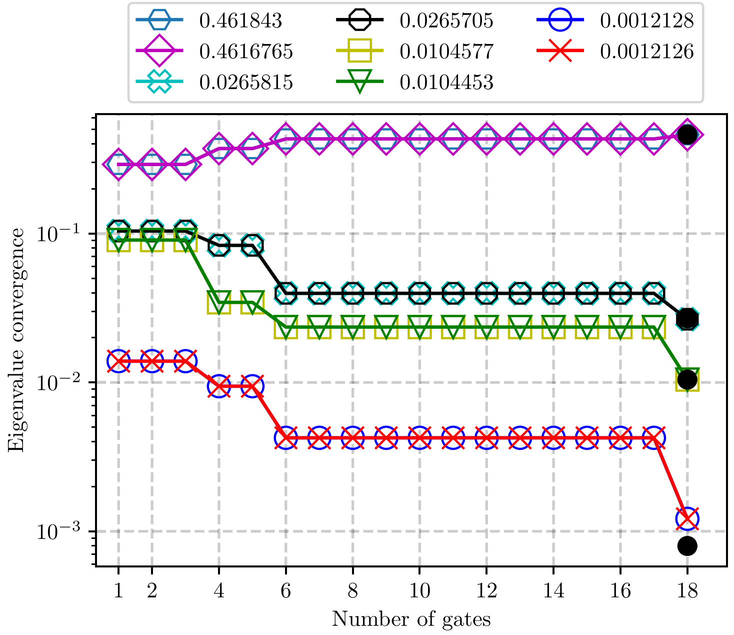


Figure 9. Convergence of the eigenvalues of the reduced ground state of 3-qubit Heisenberg model by RL-VQSD. The labels on the top of the figure correspond to the different eigenvalues. The black dots in the plot represent the true eigenvalues. There are eight black dots. However, as some of the eigenvalues coincide up to three decimal places, they are indistinguishable.

4.5. 3-qubit reduced Heisenberg model

One of the important applications of VQSD is to study the entanglement in condensed matter systems [36]. Hence, in this experiment, to get a better understanding of the efficacy of our method in this regard, we consider a 3-qubit reduced state of the ground state ($|\psi_{S_1,S_2}\rangle$) of the one-dimensional Heisenberg model defined on six qubits which have the following form

$$H = \sum_{j=1}^{2N} \vec{S}^{(j)} \cdot \vec{S}^{(j+1)}, \tag{11}$$

where $\vec{S}^{(j)} = \frac{1}{\sqrt{3}}\left(X^{(j)}\hat{x} + Y^{(j)}\hat{y} + Z^{(j)}\hat{z}\right)$ with periodic boundary condition $\vec{S}^{(2N+1)} = \vec{S}^{(1)}$, where X, Y, and Z are the Pauli operators. To perform entanglement spectroscopy on the ground state of the 6-spin Heisenberg model (i.e. $2N = 6$), we diagonalize the reduced state $\rho_{\text{red}} = \text{Tr}_{S_2}\left[|\psi_{S_1,S_2}\rangle\langle\psi_{S_1,S_2}|\right]$. We set the predefined threshold $\zeta = 10^{-4}$. We decided to choose a higher value of ζ compared to the value considered for the 2-qubit problem because as we increase the number of qubits, the problem of diagonalizing quantum states becomes more difficult, leading to complicated structures of RL-ansatz. Hence, we can choose a higher value of ζ to lower the difficulty. In appendix B, we elaborate on how the number of gates and the depth of the RL-ansatz varies as we make the problem more difficult by lowering the ζ.

The results presented in figure 9 confirm that the RL-agent can learn to construct an ansatz to find all the eigenvalues with reasonable accuracy. In this case, one can see that the ansatz takes 18 quantum gates to give us 6 out of 8 exact eigenvalues of a 3-qubit Heisenberg model. Additionally, the RL-ansatz finds the remaining two smallest eigenvalues with 1.73×10^{-7} accuracy. In figure 10, we present the RL-ansatz that contains 10 rotations and 8 `CNOT` gates proposed by the RL-VQSD. In the table 3, we investigate the performance of RL-ansatz (proposed by the RL-agent) and 4 layers of LHEA (used in VQSD) to solve 3-qubit Heisenberg model. As the metrics for the comparison, we evaluate the minimum statistical error in estimating eigenvalues, the minimum count of one and two-qubit gates, the minimum depth, and the overall minimum count of gates. It can be seen that the RL-ansatz can give us 10 times lower energy compared to LHEA with 4 layers. Meanwhile, the RL-ansatz comprises more than 3 times fewer parameters to achieve this accuracy. This clearly shows that the RL-ansatz is more efficient than the LHEA in the VQSD task and returns a smaller error in eigenvalue estimation. We also see that for depth 21 LHEA (which corresponds to the layer 1 with 39 gates), the average error in eigenvalue reaches 4.59×10^{-4}. On the other hand, for the same depth, the RL-ansatz can achieve an error of 2.43×10^{-5} with an average (over all the successful episodes) of 8 two-qubit gates and 14 one-qubit gates.

It should be noted from circuits in figure 7 and in figure 10 that the rotation in the `Z` direction, i.e. `RZ` quantum logic gate, does not play a crucial part in the diagonalizing unitary. Thus, one might attempt to

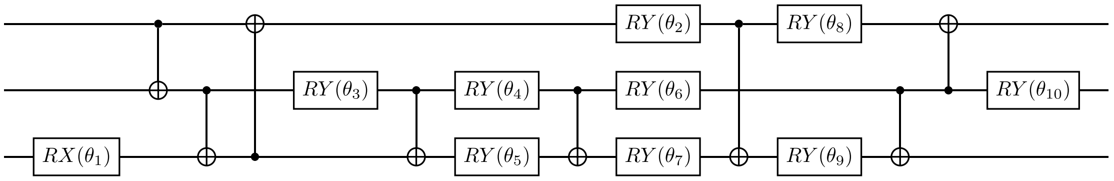


Figure 10. The ansatz proposed by the RL-agent for diagonalizing a state in the 3-qubit reduced Heisenberg model. The circuit contains 10 rotations and 8 CNOT gates.

Table 3. Comparison of the RL-ansatz (proposed by RL-agent) with 4 layers of LHEA (utilized in VQSD) used for diagonalizing 3-qubit Heisenberg model. For the comparison, we consider investigating the minimum error in eigenvalue estimation (Min. error), the minimum number of single-qubit gates (Min. 1q gate), the minimum number of the two-qubit gate (Min. 2q gate), the minimum depth (Min. depth) of the ansatz and the minimum number of total gates (Min. total gate). N.A. denotes not applicable.

	Layer	Min. error	Min. 1q gate	Min. 2q gate	Min. depth	Min. total gate
RL-ansatz	N.A.	2.43×10^{-5}	10	8	12	18
4 layers of LHEA	1	4.59×10^{-4}	36	3	21	39
	2	3.63×10^{-4}	72	6	42	78
	3	4.35×10^{-4}	108	9	63	117
	4	5.82×10^{-4}	144	12	84	156

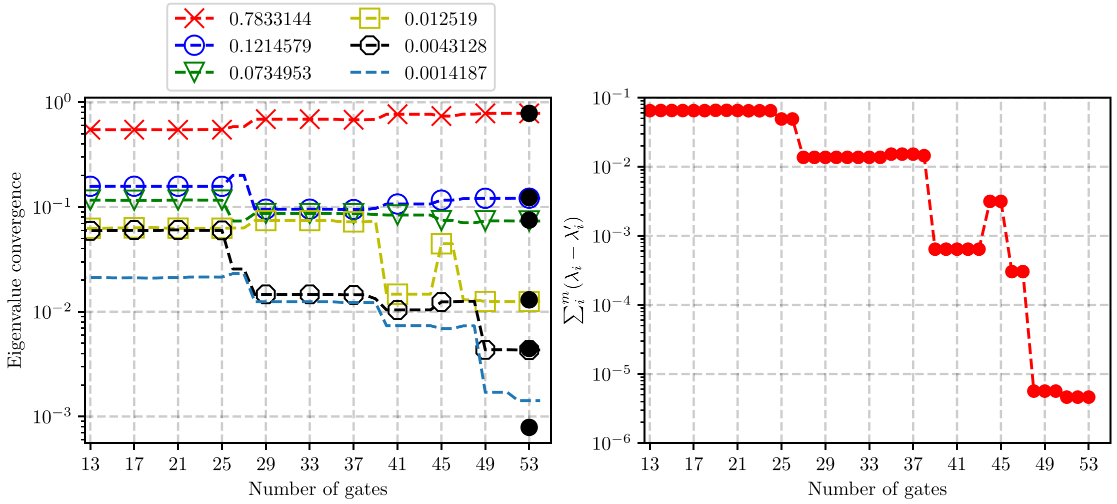


Figure 11. The convergence of individual (left panel) and the overall error (right panel) in the estimation of eigenvalues for the reduced ground state of 4-qubit Heisenberg model. This provides a significant improvement in terms of gate count and depth compared to the result reported in [1].

diagonalize a random quantum state of two and three qubits, excluding RZ rotation from the list of quantum gates. This gives us a hint concerning the action space that could be significantly reduced in these examples.

4.6. 4-qubit reduced Heisenberg model

We extend the results of the previous section for the ground state of 8-spin Heisenberg model (i.e. $2n = 8$). We diagonalize the 4-qubit reduced state of the ground state of the 8-spin Heisenberg model.

The convergence of the eigenvalues is illustrated in the figure 11. For our investigation, we show that it takes 53 gates to find the first 6 largest eigenvalues with an error below 10^{-5}. Out of 53 gates, 16 are CNOT, and the remaining are 1-qubit rotations. Throughout the experiment, we consider choosing the predefined threshold $\zeta = 10^{-3}$.

The summary of our results is provided in table 4. One can notice that there is a relation between the number of CNOTs and the dimension of the state that we want to diagonalize. The number of CNOTs grows exponentially with the number of qubits. As for the two-qubit case, we find all the eigenvalues with 10^{-10} error with just two CNOTs. Whereas for three qubits, we can find the first 6 eigenvalues with an error below 10^{-8} but the smallest two eigenvalues we find with 1.73×10^{-7} error with 8 CNOTs. Finally, for 4-qubit, we see the first 6 eigenvalues with an error below 10^{-8} and the remaining eigenvalues with an error in the range

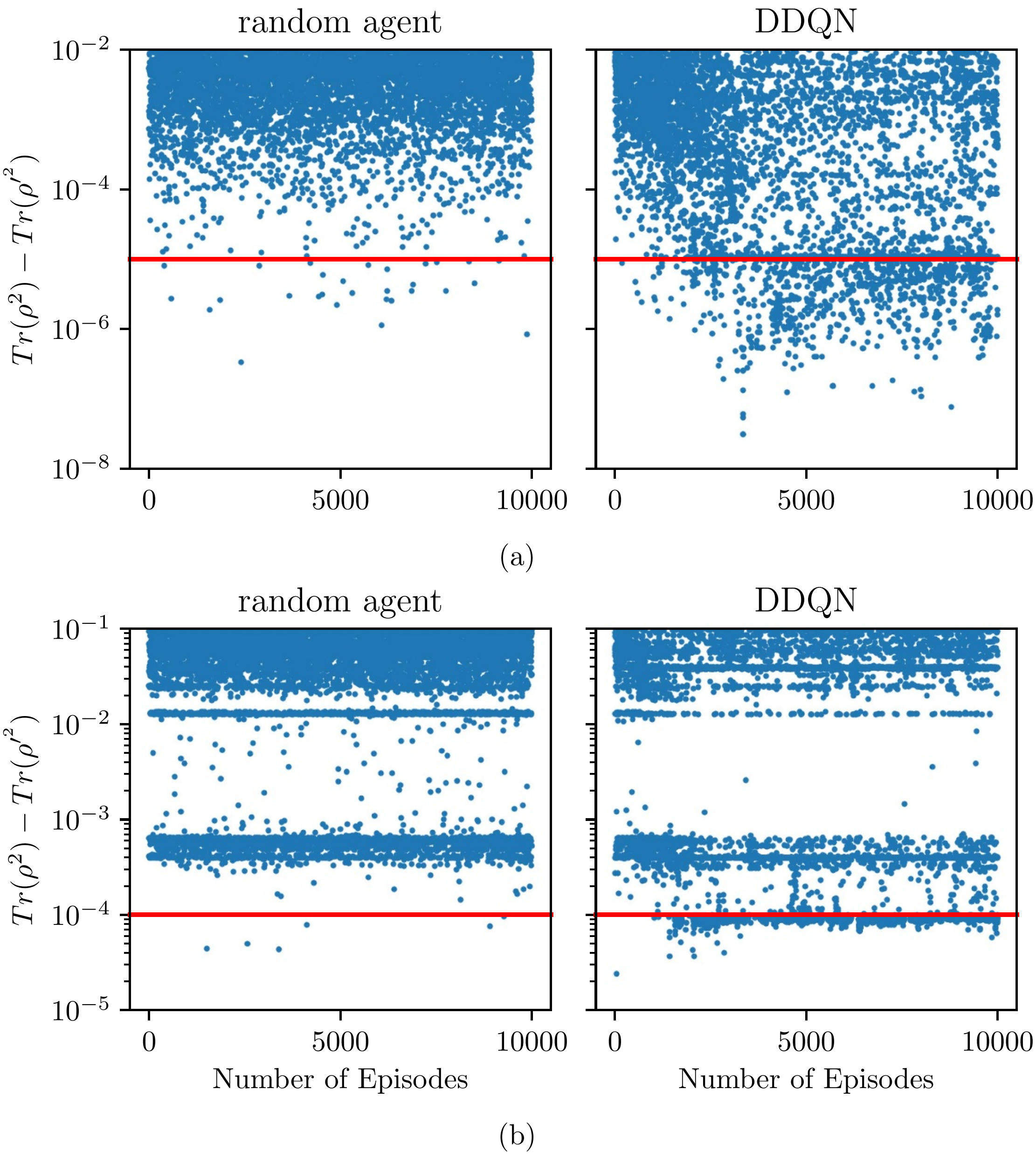


Figure 12. Comparison of accuracy obtained using a random agent and RL-based method in diagonalizing 2 and 3-qubit states. For this simulation, we utilized 10^4 episodes to solve the full rank (a) random quantum state of 2-qubit and (b) the reduced ground state of 3-qubit Heisenberg model. It can be seen that the RL-agent can give us more frequent solutions, whereas the random agent can hardly solve the problem. The red line denotes the predefined tolerance for the approximation of the cost function.

Table 4. Summary of the required minimum number of one (min. 1q gate), the minimum number of two-qubit gates (min. 2q gate) required, and the depth (min. depth) in RL-ansatz to diagonalize 2-, 3- and 4-qubit systems. To gather this data we run 10^4 episodes of the RL-VQSD for each qubit case utilizing the settings provided in setup and experimental details in the first two paragraphs of section 4.

Qubits	Min. depth	Min. 1q gate	Min. 2q gate
2	8	9	2
3	12	10	8
4	33	28	16

$10^{-4} \leqslant \Delta \leqslant 10^{-6}$ with 16 CNOTs. This observation suggests that for a full-rank quantum state of $N \geqslant 3$, we require at least as many CNOTs as the rank of the quantum state to get a good approximation of the largest eigenvalues. It should be noted that to find the first 5 largest eigenvalues with error 10^{-5} the ansatz proposed by the RL-agent is of depth 18 and a total of 30 gates, among which 12 are CNOT gates and the remaining are rotations. This significantly improves the depth, and the gate count in the diagonalizing ansatz compared to the results in [1, 5].

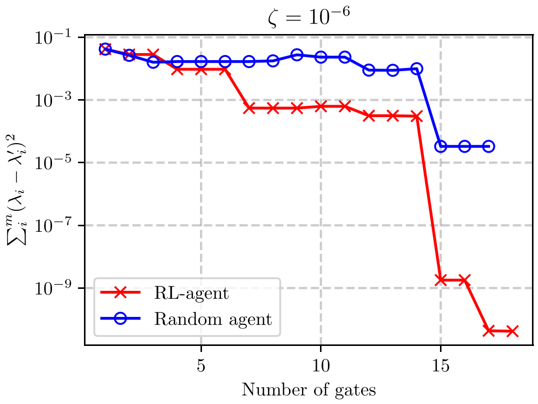


Figure 13. The comparison reaching accuracy in the order of 10^{-6} while diagonalizing 2-qubit state by RL-agent and the random agent as a function of number of gates in the circuit. To get the result we illustrate the variation of error in eigenvalue estimation with respect to the number of gates. It can be seen that the random agent halts after a certain error in eigenvalue estimation, whereas the RL-agent can go below the 10^{-6} in fewer gates.

4.7. Performance of random agent

To demonstrate the hardness of the variational diagonalization task, we utilize a random agent to find an efficient ansatz in this section. Unlike the previous examples where an RL-agent selects an action based on a policy, here in the random agent settings, the action at each step is chosen randomly from a uniform distribution.

In figure 12 (in the first column), we show the results for a random agent to diagonalize a 2- and 3-qubit quantum state. It can be seen that the number of successful episodes (the episodes that pass the predefined tolerance of cost function) drastically reduces as we scale the number of qubits in the state. At the same time, the RL-agent (in the second column) provides us with a more consistent outcome. This occurs because in the scenario of a random agent, even though the RL process is active, the NN do not utilize the information (the reward) from the environment to determine the subsequent actions. Whereas in the case of the RL-agent each subsequent action is decided based on the cumulative reward received from the environment after each step. In appendix C we investigate the training time for the RL-agent and show that the time it takes to complete an equal number of episodes by a random agent and an RL-agent is comparable.

Additionally, from the results presented in figure 13, one can conclude that even in the successful episodes, the number of gates in the ansatz proposed by the random agent is longer compared to RL-ansatz. Hence, one can argue that a random agent cannot be reliably utilized to find an efficient ansatz for the VQSD and a higher level of sophistication in learning is necessary to attain consistent results.

5. Final remarks

This paper proposed a novel method to construct the ansatz for the VQSD based on RL and compared its performance with the conventional fixed-depth ansatz. To this end, we introduced an RL-based algorithm that utilizes a novel binary encoding scheme and a dense reward function with the particular problem in mind. We showed that in solving the diagonalization problem the combination of the binary encoding and the dense reward function outperforms the previously proposed encoding and rewards proposed for solving quantum chemistry problems [8]. Indicating that proper engineering of the reward function and an efficient RL-state encoding is responsible for the agent's success. In particular, we show, for the VQSD task, a DDQN algorithm with ϵ-greedy policy can be utilized to construct an ansatz (which is termed *RL-ansatz*), shorter than the standard LHEA. As such, compared to LHEA, the RL-ansatz is of smaller depth and a smaller gate count with better accuracy in the eigenvalue estimation. This makes RL-ansatz more suitable for implementation in near-term quantum devices. Hence, the provided numerical results suggest that our approach is suitable for improving the readiness of quantum computers in tasks related to quantum data processing. The proposed state encoding method and the reward function can be readily adapted to address various variational quantum algorithms. It should be emphasized that like to emphasize that the RL-VQSD does not depend on the system size, and in principle can be used to diagonalize larger systems. However, we are currently limited by classical simulation capabilities and existing quantum devices. This is because the VQSD requires $2n$ number of qubits for n size quantum state and the training time required to get the optimal ansatz increases rapidly with system size.

Additionally, we demonstrated the hardness of the diagonalization task by replacing the RL-agent with a random agent, where the actions are chosen randomly from a uniform distribution. The results indicate that we can not reliably utilize a random agent in the diagonalization task as the number of successful episodes, where the cost function passes a predefined threshold, reduces rapidly as we scale up the size of the quantum state. Moreover, in the successful episodes, the random agent produces lengthy circuits compared to RL-agent.

To summarize our contribution, we opened up the possibility of utilizing RL to explore the quantum state diagonalization problem. Compared to the previous works on VQSD, we show that RL can boost the performance of this procedure by reducing the number of gates in the diagonalizing ansatz. As such, it provides a viable method for increasing the readiness of the VQSD algorithm for implementation on near-term quantum computers. The possibility of harnessing the cost function landscape using other search algorithms remains an open problem.

Data availability statement

The data that support the findings of this study are openly available in [34].

Acknowledgments

AK would like to thank Aritra Sarkar for the fruitful discussions. A K and J M would like to acknowledge support from the Polish National Science Center under the Grant Agreement 2019/33/B/ST6/02011, and M O would like to acknowledge support from the Polish National Science Center under the Grant Agreement 2020/39/B/ST6/01511 and from Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) programme. This research was partially supported by PL-Grid Infrastructure Grant Number PLG/2023/016205. V D was supported by the Dutch Research Council (NWO/OCW), as part of the Quantum Software Consortium programme (Project Number 024.003.037). This work was also supported by the Dutch National Growth Fund (NGF), as part of the Quantum Delta NL programme. J M would like to thank Izabela Miszczak for proofreading the manuscript.

Conflict of interest

The authors have disclosed that they do not have any competing financial or non-financial interests.

Author contributions

A K, O D, P B, and Y J P implemented the accompanying code. A K, P B, and J A M did the analysis. M B, J A M, and V D are responsible for the supervision, methodology, visualization, and preparation of the final manuscript. A K is responsible for the conceptualization.

Appendix A. Double deep Q-network

Deep RL methods employ NN to adapt the agent's policy for optimizing the return

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1},$$

with the discount factor $\gamma \in [0,1)$. Each state and action pair (s,a) can then be assigned an action-value that quantifies the expected return from state s in step t taking action a under policy π

$$q_\pi\left(s,a\right) = \mathbb{E}_\pi\left[G_t \mid s_t = s, a_t = a\right].$$

The aim is to find the optimal policy that maximizes the expected return. Such a policy can be derived from the optimal action-value function q_*, defined by the Bellman optimality equation:

$$q_*\left(s,a\right) = \mathbb{E}\left[r_{t+1} + \max_{a'} q_*\left(s_{t+1},a'\right) \mid s_t = s, a_t = a\right].$$

Instead of directly solving the Bellman optimality equation, in value-based RL, the aim is to learn the optimal action-value function from data samples. One such prominent value-based RL algorithms is Q-learning, where each state-action pair (s,a) is assigned a so-called Q-value $Q(s,a)$ which is updated to approximate q_*. Starting from randomly initialized values, the Q-values are updated according to the following rule:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left(r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)\right),$$

where α is the learning rate, r_{t+1} is the reward at time $t+1$, and s_{t+1} is the next encountered state after taking action a_t in state s_t. In the limit of visiting all (s,a) pairs infinitely often, this update rule is proven to converge to the optimal Q-values in the tabular case [37]. In practice, to ensure sufficient exploration in Q-learning setting, a so-called ϵ-greedy policy is used. Formally, stated as,

$$\pi(a \mid s) = \begin{cases} 1-\epsilon_t \, for \; a = \max_{a'} Q(s,a') \\ \epsilon_t \; otherwise. \end{cases}$$

The ϵ-greedy policy is only used to introduce randomness to the actions selected by the agent during training, but once training is finished, a deterministic policy follows.

We employ NN as function approximators to extend Q-learning to large state and action spaces. NN training typically requires independently and identically distributed data, which is not naturally available in the sequential RL data. This problem is circumvented by experience replay. This method divides past experiences into single-episode updates, creating batches that are randomly sampled from a memory. To stabilize training, two NNs are employed, a policy network, that is continuously updated and a target network that is an earlier copy of the policy network. The policy network estimates the current value, while the target network provides a more stable target value, represented by Y:

$$Y_{\mathrm{DQN}} = r_{t+1} + \gamma \max_{a'} \mathrm{Q}_{\mathrm{target}}(s_{t+1},a')$$

In the DDQN algorithm, the action for the target value is sampled from the policy network to reduce the overestimation bias inherent in standard DQN. The corresponding target is defined as:

$$Y_{\mathrm{DDQN}} = r_{t+1} + \gamma \mathrm{Q}_{\mathrm{target}}\left(s_{t+1}, \arg\max_{a'} \mathrm{Q}_{\mathrm{policy}}(s_{t+1},a')\right).$$

This target value is approximated using a selected loss function, in this case, a smooth L1-norm loss.

Appendix B. Dependency of gates and depth on predefined threshold

Throughout the paper, we have chosen the predefined threshold ζ constant for a fixed problem. For example, while solving two-qubit random states we choose $\zeta = 10^{-5}$, which is increased to $\zeta = 10^{-4}$ and later $\zeta = 10^{-3}$, for the task of diagonalizing a 3 and 4-qubit Heisenberg model respectively. Here, we investigate the dependency of the number of gates and the depth of an RL-ansatz for a varying ζ. It is straightforward to understand that the lower the ζ, the more difficult it is to solve the diagonalizing problem, as a lower threshold corresponds to higher accuracy in eigenvalue estimation. Hence, we expect to observe an apparent increase in the number of gates and depth of the circuit as the threshold moves towards a lower value.

The results are summarized in table 5 where we consider the RL-VQSD to diagonalize the 3-qubit Heisenberg model while the ζ is set from 10^{-3} to 10^{-9} in an interval of 10^{-2}. We see that the number of gates in the circuit and the depth increase gradually as we lower the threshold.

Table 5. We summarize the influence of the threshold on the number of gates and depth of the RL-ansatz. To gather data, we run 3000 episodes of RL-VQSD to solve the 3-qubit Heisenberg model, and the results are averaged over all the successful episodes.

ζ	Avg. 1q gate	Avg. 2q gate	Avg. depth	Avg. num gate
10^{-3}	13.46	9.62	16.63	23.08
10^{-5}	17.24	10.62	19.67	27.86
10^{-7}	21.10	20.03	31.63	41.13
10^{-9}	25.30	20.85	36.80	46.15

Table 6. The details of GPU and CPU resources utilized to record the training time.

CPU	`Intel(R) Core(TM) i7-10700KF CPU @ 3.80GHz`
GPU	`NVIDIA GA102 [GeForce RTX 3080 Ti] 64 bits`

Table 7. The record of the training time it takes for the RL-agent and the time it takes for the random agent to complete the same number of episodes in diagonalizing a 2 and 3-qubit state. As for the 2-qubit, we choose an arbitrary full-rank state, and for 3-qubit, we consider the reduced ground state of the Heisenberg model. The time is recorded for 3000 episodes, which is more than sufficient to solve the diagonalization problem with the predefined threshold in the range 10^{-4} to 10^{-5}.

System	Method	Time per episode (in seconds)	Total time (in hours)
2 qubit	RL-agent	9.21	11.45
	Random agent	9.29	11.15
3 qubit	RL-agent	37.74	31.45
	Random agent	37.70	31.42

Appendix C. Training time

Here we discuss the time it takes to train the RL-agent in diagonalizing 2- and 3- qubit states. To record the time we run the RL-VQSD algorithm to diagonalize 2-qubit arbitrary quantum state and the reduced ground state of 3-qubit Heisenberg model for 3000 episodes. The details of the CPU and GPU that are utilized to gather data are provided in table 6. To gain more insight in table 7 we compare the training time of the RL-agent with the time it takes to complete an equal amount of episodes by a random agent setting and show that in case of diagonalizing the 2- and 3-qubit both the methods takes the same amount of time.

ORCID iDs

Akash Kundu https://orcid.org/0000-0002-3540-1061
Mateusz Ostaszewski https://orcid.org/0000-0001-7915-6662
Onur Danaci https://orcid.org/0000-0002-7320-8297
Vedran Dunjko https://orcid.org/0000-0002-2632-7955
Jarosław A Miszczak https://orcid.org/0000-0001-8790-101X

References

[1] LaRose R, Tikku A, O'Neel-Judy E, Cincio L and Coles P J 2019 Variational quantum state diagonalization *npj Quantum Inf.* **5** 57
[2] Cerezo M, Poremba A, Cincio L and Coles P J 2020 Variational quantum fidelity estimation *Quantum* **4** 248
[3] Kundu A and Adam Miszczak J A 2022 Variational certification of quantum devices *Quantum Sci. Technol.* **7** 045017
[4] Zeng J, Cao C, Zhang C, Xu P and Zeng B 2021 A variational quantum algorithm for Hamiltonian diagonalization *Quantum Sci. Technol.* **6** 045009
[5] Cerezo M, Sharma K, Arrasmith A and Coles P J 2022 Variational quantum state eigensolver *npj Quantum Inf.* **8** 113
[6] Lloyd S, Mohseni M and Rebentrost P 2014 Quantum principal component analysis *Nat. Phys.* **10** 631–3
[7] Cincio L, Y Subaşi, A T Sornborger and Coles P J 2018 Learning the quantum algorithm for state overlap *New J. Phys.* **20** 113022
[8] Ostaszewski M, Trenkwalder L M, Masarczyk W, Scerri E and Dunjko V 2021 Reinforcement learning for optimization of variational quantum circuit architectures *35th Conf. on Neural Information Processing Systems* pp 18182–94
[9] Kuo E-J, Fang Y-L L and Chen S-Y 2021 Quantum architecture search via deep reinforcement learning (arXiv:2104.07715)
[10] Ye E and Yen-Chi Chen S 2021 Quantum architecture search via continual reinforcement learning (arXiv:2112.05779)
[11] He Z, Zhang X, Chen C, Huang Z, Zhou Y and Situ H 2023 A GNN-based predictor for quantum architecture search *Quantum Inf. Process.* **22** 128
[12] Bolens A and Heyl M 2021 Reinforcement learning for digital quantum simulation *Phys. Rev. Lett.* **127** 110502
[13] Zhang S-X, Hsieh C-Y, Zhang S and Yao H 2022 Differentiable quantum architecture search *Quantum Sci. Technol.* **7** 045023
[14] Du Y, Huang T, You S, Hsieh M-H and Tao D 2022 Quantum circuit architecture search for variational quantum algorithms *npj Quantum Inf.* **8** 62
[15] Patel Y J, Jerbi S, Bäck T and Dunjko V 2022 Reinforcement learning assisted recursive qaoa (arXiv:2207.06294)

[16] Moro L, Paris M G A, Restelli M and Prati E 2021 Quantum compiling by deep reinforcement learning *Commun. Phys.* **4** 178

[17] Anonymous 2023 Curriculum reinforcement learning for quantum architecture search under hardware errors *Submitted to The Twelfth Int. Conf. on Learning Representations* (available at: https://openreview.net/forum?id = rINBD8jPoP)

[18] Demmel J W, Marques O A, Parlett B N and Vömel C 2008 Performance and accuracy of LAPACK's symmetric tridiagonal eigensolvers *SIAM J. Sci. Comput.* **30** 1508–26

[19] Powell M J D 1994 A direct search optimization method that models the objective and constraint functions by linear interpolation *Advances in Optimization and Numerical Analysis* (*Mathematics and Its Applications*) vol 275, ed S Gomez and J P Hennart (Springer)

[20] Powell M J D 2006 A fast algorithm for nonlinearly constrained optimization calculations *Numerical Analysis: Proc. Biennial Conf. Held at (Dundee, 28 June–1 July 1977)* (Springer) pp 144–57

[21] Baumgratz T, Cramer M and Plenio M B 2014 Quantifying coherence *Phys. Rev. Lett.* **113** 140401

[22] Peruzzo A, McClean J, Shadbolt P, Yung M-H, Zhou X-Q, Love P J, Aspuru-Guzik A and O'Brien J L 2014 A variational eigenvalue solver on a photonic quantum processor *Nat. Commun.* **5** 4213

[23] Farhi E, Goldstone J and Gutmann S 2014 A quantum approximate optimization algorithm *Technical Report* MIT-CTP/4610 (Massachusetts Institute of Technology)

[24] Filip M-A and Thom A J W 2020 A stochastic approach to unitary coupled cluster *J. Chem. Phys.* **153** 214106

[25] Taube A G and Bartlett R J 2006 New perspectives on unitary coupled-cluster theory *Int. J. Quantum Chem.* **106** 3393–401

[26] Bartlett R J, Kucharski S A and Noga J 1989 Alternative coupled-cluster ansätze II. The unitary coupled-cluster method *Chem. Phys. Lett.* **155** 133–40

[27] Sutton R S and Barto A G 2018 *Reinforcement Learning: An Introduction* (MIT Press)

[28] Van Hasselt H, Guez A and Silver D 2016 Deep reinforcement learning with double q-learning *Proc. AAAI Conf. on Artificial Intelligence* vol 30

[29] Mnih V *et al* 2015 Human-level control through deep reinforcement learning *Nature* **518** 529–33

[30] Fösel T, Yuezhen Niu M, Marquardt F and Li L 2021 Quantum circuit optimization with deep reinforcement learning (arXiv:2103.07585)

[31] Guo C and Berkhahn F 2016 Entity embeddings of categorical variables (arXiv:1604.06737)

[32] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D and Riedmiller M 2013 Playing atari with deep reinforcement learning *Advances in Neural Information Processing Systems* (Deep Learning Workshop)

[33] Kingma D P and Ba J 2015 Adam: a method for stochastic optimization *3rd Int. Conf. for Learning Representations*

[34] Code for "Enhancing quantum variational state diagonalization using reinforcement learning techniques" (available at: https://github.com/iitis/RL_for_VQSD_ansatz_optimization)

[35] Aleksandrowicz G *et al* Qiskit: an open-source framework for quantum computing (available at: https://qiskit.org/)

[36] Li H and Haldane F D M 2008 Entanglement spectrum as a generalization of entanglement entropy: identification of topological order in non-abelian fractional quantum hall effect states *Phys. Rev. Lett.* **101** 010504

[37] Melo F S 2001 Convergence of q-learning: a simple proof *Technical Report* (Institute Of Systems and Robotics) pp 1–4