**PAN: Pulse Ansatz on NISQ Machines**

Zhiding Liang∗†, Jinglei Cheng†‡, Hang Ren†, Hanrui Wang§, Fei Hua∥, Yongshan Ding∥∥
Fred Chong∗∗, Song Han††, Yiyu Shi*, Xuehai Qian††

*University of Notre Dame †Purdue University ‡University of California, Berkeley
§Massachusetts Institute of Technology ∥University of Rutgers
∥∥Yale University **University of Chicago

†These authors contributed to the work equally and should be regarded as co-first authors.

Corresponding authors: zliang5@nd.edu

**Abstract—** Variational quantum algorithms (VQAs) have demonstrated great potentials in the NISQ era. In the workflow of VQA, the parameters of ansatz are iteratively updated to approximate the desired quantum states. We have seen various efforts to draft better ansatz with less gates. Some works consider the physical meaning of the underlying circuits, while others adopt the ideas of neural architecture search (NAS) for ansatz generator. However, these designs do not exploit full advantages of VQA. Because most techniques are targeting gate ansatz, and the parameters are usually rotation angles of the gates. In quantum computers, the gate ansatz will eventually be transformed into control signals such as microwave pulses on transmons. And the control pulses need elaborate calibration to minimize the errors such as over-rotation and under-rotation. In the case of VQAs, this procedure will introduce redundancy, but the variational properties of VQAs can naturally handle problems of over-rotation and under-rotation by updating the amplitude and frequency parameters. Therefore, we propose PAN, a native-pulse ansatz generator framework for VQAs. We generate native-pulse ansatz with trainable parameters for amplitudes and frequencies. In our proposed PAN, we are tuning parametric pulses, which are natively supported on NISQ computers. Considering that parameter-shift rules do not hold for native-pulse ansatz, we need to deploy non-gradient optimizers. To constrain the number of parameters sent to the optimizer, we adopt a progressive way to generate our native-pulse ansatz. Experiments are conducted on both simulators and quantum devices to validate our methods. When adopted on NISQ machines, PAN obtained improved the performance with decreased latency by an average of 86%. PAN is able to achieve 96.482% and 99.336% accuracy for VQE tasks on H2 and HeH+ respectively. An average accuracy of 97.27% is achieved for medium-size VQE tasks on CO2, H2O, and NaH. PAN also demonstrates advantages on QAOA tasks even with considerable noises in NISQ machines.

I. INTRODUCTION

Operating on the principles of quantum mechanics, quantum computers have the potentials to solve problems that are intractable on classical computers [4], [12], [44]. As the hardware technologies and quantum algorithms advance rapidly, today’s quantum computers begin to demonstrate their advantages in solving problems of non-trivial size in areas such as quantum chemistry [9]. In 2019, Google claimed to have achieved quantum supremacy with the task of random circuit sampling with a 53-qubit quantum computer [8]. IBM launched its 127-qubit quantum computer at the end of 2021, together with a roadmap for quantum processors with more than 1000 qubits in the coming years [1]. In the current NISQ era, however, emerging quantum devices are still prone to errors and sensitive to decoherence [26], [41], [79].

In NISQ computers, qubits are scarce and not perfectly isolated from the environments. These quantum devices have not yet met the hardware requirements for error correction methods such as surface codes [22], [23], [27], [40], [68] due to the limited number of qubits and low gate fidelity. Without quantum error correction, current quantum computers can only handle small-scale circuits before running into irreversible errors, making practical-size algorithms infeasible. However, with elaborately designed noise-resilient algorithms, we can still expect to achieve quantum supremacy in areas such as quantum chemistry [16], [47], [67], [72] much sooner than other applications like database search [37] and integer factorization [65].

Variational quantum algorithms [7], [17], [38], [51], [66], [76] have shown great noise resilience, and are considered as hybrid algorithms, where some parts are performed on a quantum device and others on a classical computer. Variational quantum eigensolver (VQE) [43], [50], [74] is one of the most promising candidates in the variational computing paradigm. With VQE, we are able to estimate the ground state energy of a targeting quantum system by iteratively evaluating and updating a parametrized quantum circuit. Quantum approximate optimization algorithm (QAOA) [38] and quantum neural network (QNN) [3] are also members of VQAs. QAOA attempts to solve combinatorial optimization problems including max-cut problem. And QNN has exhibited exceptional capabilities to represent complex data [39], [63]. These algorithms are among the most promising examples of NISQ algorithms since the number of required quantum gates remain moderate.

**Exposing Native Pulse-Level Controls.** The majority of existing quantum computers do not provide access for analog controls of qubits. Consequently, nearly all compilers implement a gate-based workflow [48], [71], in which quantum algorithms are synthesized, compiled on classical computers, and finally executed on quantum computers. In the first place, quantum circuits are generated or synthesized to implement certain functions of quantum algorithms. These gate circuits are usually not compatible with the underlying topology of quantum hardware. Therefore, SWAP gates need to be inserted to make quantum circuits executable on quantum computers. Then these circuits are decomposed into single-qubit and two-qubit gates that are natively supported by quantum computers.
Finally the circuits are dispatched to quantum backends, where they are “translated” into control signals on physical qubits such as transmons [61], trapped ions [13] and photons [45]. For superconducting quantum computers, the control signals are microwave pulses [6].

There is more flexibility if gate circuits are decomposed and controlled at pulse level, since we are dealing with a more fine-grained abstraction layer. The challenge now is to figure out how to effectively generate pulses for quantum algorithms. Existent techniques such as quantum optimal control (QOC) [19], [54], [78] can be adopted to generate control pulses. QOC devises and implements the shapes of external controls on qubits to accomplish given tasks. As indicated in [64], QOC can handle quantum circuits of moderate size, but the scalability might be the issue. Despite various efforts [18], [46] to optimize QOC and reduce its overhead, it is still computationally expensive. Besides, we need to consider the high-noise feature of NISQ devices, on which it is preferred to have a fixed set of allowed operations. On current quantum computers, a small group of gates are carefully calibrated regularly to maintain their accuracies. Overall, it is generally hard to take advantage of quantum pulses with NISQ machines.

Why Pulses for VQAs? Situations differ slightly in the case of VQAs due to its variational characteristics. During the “training” process, the parametric circuits of VQAs are updated iteratively. It is now unimportant whether the controls are accurately implemented on quantum hardware as long as the parametric circuits can reach the desired states. Instead of using gate-level compilation or QOC, we can implement quantum algorithms at the “native-pulse level”. At the native-pulse level, we can directly manipulate the native pulses that are supported by the quantum hardware. This paradigm change grants more fine-grained control and thus making it possible for better performance, scalability, and robustness. A recent work [32] has shed light on the feasibility and potentials for such a paradigm change. However, the research on pulse level optimization is still in its infancy. The capability of quantum pulses has not been fully explored, nor are they already robust or scalable on NISQ machines. As summarized in Table I, many critical issues remain unsolved.

To tackle these challenges, we propose PAN, a native-pulse ansatz generator for VQAs. PAN is the first to demonstrate the feasibility of native-pulse ansatz on NISQ machines. Instead of using rotation gates, we directly use the pulses that are natively supported by the quantum processors. Compared with QOC, PAN has fewer parameters and can be easily deployed onto NISQ machines, while QOC with realistic system models requires huge computation resources. On the other side, PAN is superior to gate-based methods, since PAN drops the abstraction layer of native gates and results in less circuit latency. We provide results from IBM’s superconducting quantum computers [52], while previous pulse-level works are only evaluated on simulators.

Contributions. The goal of this paper is to construct native-pulse ansatz for VQAs and demonstrate in both simulators and NISQ machines. The major contributions of PAN include:

- **Native-pulse ansatz.** Our native-pulse ansatz is derived from native pulses that are extracted from quantum backends. In this way, we ensure that pulse ansatz is compatible with quantum hardware.
- **Progressive learning.** In PAN, a non-gradient optimizer is employed. We provide a progressive way to “grow” our native-pulse ansatz in order to maintain a reasonable size for the parameters handled by the optimizer. New pulse blocks with zero amplitudes are appended at different “steps” of PAN. This prevents the appended pulses from abruptly changing the ansatz circuits’ overall unitaries.
- **Results from NISQ machines.** Experiments are conducted on simulators and NISQ machines. The results show that the native-pulse ansatz outperforms the gate ansatz for VQA tasks in terms of accuracy and latency.
- **Exploration on frequency tuning.** We explore possible benefits of tuning pulse frequencies on transmons. Experimental results show that pulse frequency can be an extra degree of freedom for native-pulse ansatz.

Evaluation Highlights. Six NISQ machines are used to validate PAN. We achieve latency reductions of up to 97.3% compared to baselines. Accuracies up to 99.895% are attained for small-size VQE tasks.

II. BACKGROUND

A. Qubit and Quantum Gate.

In classical computing, the basic element to represent data is a classical bit that can be 0 or 1. Similarly, in quantum computing, the basic element to represent data is a quantum bit. While classical bits are limited to the values 0 and 1, qubits can be linear combinations of two states that correspond to 0 or 1. The state of a qubit $|\psi\rangle$ can be described as a linear combination of the state 0 and state 1: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. In analogy with classical computation, qubits are manipulated via quantum gates. For instance, the X gate is used to transform the state of a qubit from $|\alpha\rangle + |\beta\rangle$ to $|\beta\rangle - |\alpha\rangle$. Quantum gates are the fundamental components of quantum circuits, which can be used to manipulate the state of qubits and implement a wide range of quantum algorithms.
TABLE I: Comparison between gate-level approaches, pulse-level approaches and the proposed PAN

| Method                          | Robustness | Parameters of Control | Applications |
|--------------------------------|------------|-----------------------|--------------|
| Gate-level (conventional)       | ✓          | ✓                     | ✓            |
| Pulse-level attempts [19], [54] | ✗          | ✗                     | ✓            |
| PAN (Proposed)                  | ✓          | ✓                     | ✓            |

B. Variational Quantum Algorithms (VQAs)

The variational quantum eigensolver (VQE) is one of the most popular and promising VQAs. VQE is employed primarily to solve the ground and low-excited states of quantum systems. In addition, it has important applications in quantum many-body physics, quantum chemistry, and other fields [43], [59], [70]. VQE has also been shown to be noise resistant in NISQ devices [74]. The Quantum Approximate Optimization Algorithm (QAOA) is a variational quantum algorithm that solves combinatorial optimization problems with sub-optimal solutions [29], [38]. It is commonly believed that QAOA can establish quantum supremacy in NISQ computers with shallow circuits. Quantum neural network (QNN) is a model of quantum machine learning (QML) that uses ansatz circuits to extract features from input data, followed by complex-valued linear transformations. QNN has great potentials for applications in QML [5], [11], quantum simulation and optimization [55].

C. Quantum Optimal Control

Assuming we have a closed quantum system, the Hamiltonian of the system is provided by

\[ H(t) = H_0 + \sum_{j=1}^{D} u_j(t) H(j) \]  

(1)

where \( H_0 \) is the drive Hamiltonian, \( H(j) \) is the control Hamiltonian, and \( u_j \) is time-dependent control signal. The Schrödinger’s equation governs the system’s dynamics:

\[ \frac{d}{dt} \psi(t) = -iH(t)\psi(t) \]  

(2)

where \( \psi(t) \) denotes the system’s state at time \( t = 0 \). Quantum optimum control (QOC) can be used to calculate \( u_j(t) \) in order to adjust the control signals. Using QOC, for instance, we can transform one quantum state into a desired state. Typically, a cost function is specified as the pulse’s fidelity, which measures the difference between the simulated unitary matrix and the desired unitary matrix. Algorithms like GRAPE [24] and CRAB [15] can be applied to generate the control signals.

D. Quantum Pulse Learning

Another method of generating pulses is to parameterize the quantum pulses and then optimize the parameters. We refer to such process as quantum pulse learning. To comprehend the trainable components of control pulses for transmon quantum computers, the Hamiltonian can be defined as follows:

\[ H = \sum_{i=0}^{1} (U_i(t) + D_i(t)) \sigma_i^X + \sum_{i=0}^{1} 2\pi \nu_i (1 - \sigma_i^Z)/2 \]

\[ + \omega_B a_B a_B^\dagger + \sum_{i=0}^{1} g_i \sigma_i^X (a_B + a_B^\dagger) \]  

(3)

where \( U_i(t) \) and \( D_i(t) \) are two major terms that govern the pulse learning, they are derived in Equation 4. They are obtained by mixing local oscillator with control signals. \( \sigma_X, \sigma_Y, \) and \( \sigma_Z \) are Pauli operators. \( \nu_i \) is the estimated frequency of qubits in the qubit \( i \), \( g_i \) is the coupling strength between qubits, \( \omega_B \) is the frequency of control buses, \( a_B \) and \( a_B^\dagger \) are the ladder operator for control buses.

\[ D_i(t) = \text{Re}(d_i(t)e^{i\omega d_i t}) \]  

\[ U_i(t) = \text{Re}[u_i(t)e^{i\omega i - \omega u_i t}] \]  

(4)

where \( d_i(t) \) and \( u_i(t) \) are the signals of qubit \( i \) on drive channel and control channel. Since pulse learning adjust \( d_i(t) \) and \( u_i(t) \), \( D_i(t) \) and \( U_i(t) \) are changed accordingly. Consequently, the drive Hamiltonian is also updated [49]. Thus, we are able to manipulate the quantum system with control signals.

E. Current NISQ machines

Several prospective material systems are being researched in order to build and implement qubits and quantum gates. Material systems include trapped ions [58], optical lattices [36], nuclear magnetic resonance (NMR) [73], diamond [56], and superconducting circuits [20]. Superconducting circuit is one of the most leading candidates to build a quantum computer of realizing the computations beyond the reach of classical computer. In superconducting transmon system, quantum gates are implemented by driving the target qubit through the microwave in the pulse level [6]. For example, the Rx and Ry rotations can be implemented by sending microwave voltage signals. Due to the imperfect implementations, efficient and precise pulse control remains as an open question, which also provides more optimization opportunities.

III. Motivation

A. Deficiencies of Gate-level Compiler

Figure 3 illustrates the execution of a quantum computing program from high-level programming to the compiler on quantum machines. Redundancy is introduced by the existing implementation of rotation gates on quantum hardware. For
Fig. 2: The overview of design and implementation of PAN. The proposed pulse ansatz is composed of single-qubit native pulses (SNP) and two-qubit native pulses (TNP). During the training process, the ansatz is “grown” after each step. In this progressive way, older parameters are unchanged, while newer parameters are trained in the next step.

Fig. 3: Comparison between compilation process for gate level and pulse level. Gate-level workflow consists of several layers and introduce redundancy, pulse-level workflow consists fewer layers that can provide advantages on latency.

example, we want the qubit to evolve to the point on the Bloch sphere as indicated in Figure 1. In order to reach such state, a basis transformation, a phase shift, and a second basis transformation must be performed. This example illustrates the redundancy when quantum circuits are compiled into native gates. We propose to bypass the abstract layer of native gates and use native pulses directly as parametric elements in the ansatz.

B. Drawbacks of Optimal Control Pulses

In addition, attempts have been made to optimize the quantum pulse generator. For example, [33], [78] propose using quantum optimal control to generate pulses for given unitaries. The applications of quantum pulses are severely constrained by the excessive cost of pulse generators and the complexity of NISQ devices’ system models. QOC is usually adopted when a target unitary or state transition is known. Hence, it is incompatible with VQAs, which lack target unitaries. In addition, QOC’s gradient-based approach with back propagation cannot be used, because gradients for pulses are not accessible on NISQ machines.

C. Potentials of Pulse Ansatz

As the illustration in Figure 1, if we can directly control pulses, we can eliminate some redundancy. This allows us to reduce the program’s overall latency. Using pulses rather than gates can result in a lack of calibration, which may result in inaccurate operations. However, for VQAs, the parameterized pulse ansatz can be automatically adjusted for errors such as under-rotation and over-rotation. Besides amplitudes of native pulses, the frequency of qubit channels can be tuned. In conclusion, pulse ansatz offers more degrees of freedom, enabling us to search for desired states with substantially shorter pulse latencies.

IV. OVERVIEW OF PAN FRAMEWORK

In the proposed framework for native-pulse ansatz, the pulse parameters determine the drive Hamiltonian in Equation 3. However, gradients are not available if we “train” the pulse ansatz on NISQ machines. Therefore, we have to use a non-gradient optimizer to train the parameters in the ansatz. To mitigate the drawbacks of non-gradient optimizer, we offer a progressive method to generate our native-pulse ansatz. Such progressive learning structure ensures that the optimizer’s parameter dimensions don’t exceed its capacity. This machine-in-loop training method also makes the framework noise-resilient. We illustrate the workflow of the proposed framework PAN in Figure 2. Firstly, the configurations of the NISQ computers are extracted. The configurations include information regarding qubit frequencies and the mapping between native gates and native pulses. After obtaining the native pulses supported by
quantum backends, we can progressively construct our native pulse ansatz. Similar to the hardware-efficient ansatz, single-qubit gates are placed for each qubit and two-qubit gates are applied to available connections. The single-qubit pulses are derived from pulses retrieved from Hadamard gate or Rx gate. And two-qubit pulses are derived from CX gate or CR gate. Pulses from CR gates are preferred because they are the simplest pulses that enable the entanglement of two qubits. In this way, we now have two types of layers. One consists of single-qubit native pulses on all qubits, whereas the other consists of two-qubit native pulses on available connections. The two types of layers are alternately inserted during the training process to help explore the Hilbert space. To train the native-pulse ansatz, we use non-gradient optimizers. The incrementally constructed ansatz prevents non-gradient optimizers from failing to work with huge dimensions.

Simulators and NISQ machines are employed to evaluate our methods. On the simulators, we present energy-distance curves for several molecules. The energy curves closely resemble those generated using the full configuration interaction (FCI) approach. The results show that our pulse ansatz can approximate the lowest energy states of molecules with much shorter durations. Our ansatzes are also tested on NISQ machines. In addition, we demonstrate that tuning frequencies on NISQ computers is beneficial.

V. DESIGN AND IMPLEMENTATION DETAILS

A. Gate Ansatz versus Native-Pulse Ansatz

In the proposed PAN, we replace the basic element of the variational quantum circuit with a native pulse and use quantum pulses to build a native-pulse ansatz. During the training process, parameters of native pulses are updated, thus obviating the requirement for decomposition into native gates. To create a native-pulse ansatz, we must guarantee that the employed pulses are parametric pulses supported by the NISQ device. Since we use native pulses as building blocks for our ansatz, we can explore all available parameters that are allowed to be tuned. In the case of gate-based ansatz, the parameters are only angles of rotation gates. These gates are decomposed into native gates before being implemented with microwave pulses and phase shifts. The microwave pulses are applied to qubits, while the phase shift is acting on classical electronics. When the angles for the rotation gates change, we observe changes in phase shift. However, the microwave pulses have not changed. Consequently, gate-based methods cannot take advantage of parametric pulses. So, we choose to directly adjust the parameters of the microwave pulses acting on qubits. In this way, we introduce opportunities to simplify the control signals, thus reducing the overall latency of quantum circuits. As shown in Figure 3, we are able to reduce the number of abstract layers. Another advantage of adjusting pulse parameters instead of gate angles is the mitigation of gate-to-pulse compilation noise. To experimentally realize a continuous parametric gate such as RX(\(\pi/4\)), the amplitude of \(X_{\pi/4}\) pulse is set to be half of that of a \(X_{\pi/2}\) pulse, i.e., \(A_{\pi/4} = A_{\pi/2}/2\). However, this step introduces noise due to the nonlinearity in NISQ devices [31]. PAN can directly tune the underlying pulse parameters and avoid such gate-to-pulse compilation noises.

B. Parameters of Native-Pulse Ansatz

Tuning amplitudes. With native-pulse ansatz, we can tune parameters that are not accessible in gate-based ansatz. For example, the amplitudes of the pulses on the drive and control channels are invisible to gate-level users. We define two types of native pulses in our framework. Single-qubit native pulses (SNPs) are based on native pulses for different one-qubit gates. And Two-qubit native pulses (TNPs) are based on native pulses for CR gate, since the CR gate is the most basic element that triggers the entanglement of two qubits. Both SNP and TNP are initialized with zeros as their amplitudes. The reason is explained in section V-D.

We can see from Eq. 3 and 4 that when we change the parameters of the pulses, it changes the strength of the signal on the control channel of \(d_i(t)\), which eventually changes the drive Hamiltonian. Experimental results presented in Table II confirm the advantages of pulse ansatz over gate ansatz. For experiments in Table II, the target is to solve the ground state energy of \(H_2\) with VQE on the NISQ machine ibmq_jakarta. By comparing the results of SNP, TNP, TNP+SNP, and the two-gate ansatz, we demonstrate that the pulse ansatz provides a better energy value for the task in 31.2% less duration. The purpose of comparing the two-gate ansatz and the TNP+SNP pulse circuit is to highlight the advantages of native-pulse ansatz construction, as CX and TNP are the simplest 2-qubit operations on the gate-level and pulse-level, respectively.

Tuning frequencies. Clifford and T gates are universal to perform arbitrary quantum operations [35], so gate calibrations mainly fine-tune a discrete set of \(\pi/2\) and \(\pi\) pulses corresponding to Clifford operations [69]. In the progressive pulse learning protocol, pulse parameters are continuously varied, so it is necessary to verify that parametric pulses still produce physical quantum operations, i.e., pulses correspond to quantum operations and can be implemented with a high fidelity [77]. We demonstrate the performance of pulse implementation by running a pulse version of the gate sequence \((CX + H + H^\dagger + CX^\dagger)\) under different frequency detuning. If the pulse block \((CX + H)\) is physically feasible to realize, when pulse detuning is zero, the circuit should produce a final \(|00\rangle\) state with high probability. Experiment results in Figure 4 show a near-to-one prob \((00)\) readout result for the zero detuning case and thus prove the feasibility of parametric pulse operations. The high-fidelity results also indicate that PAN does not introduce additional noise and can be authentically implemented.

In addition, the frequency detuned results verify that the magnitude of the detuning range we choose \((\approx 2MHz)\) is wide enough such that the variation of pulse parameters can lead to the obvious changes of the corresponding operation, and the resolution of pulse variation in the learning process is appropriate to ensure fast convergence and reduce the
TABLE II: Comparison of trainability for different pulse circuits and gate circuits on ibmq_jakarta.

| Operations | Circuit Level | Molecule Bond Length | Reference Energy | VQE ($H_2$) Result | Duration(on ibmq_jakarta) |
|------------|---------------|----------------------|------------------|---------------------|--------------------------|
| SNP        | Pulse Circuit | 0.1 Å                | 2.710H           | 4.380H              | 71.1ns                   |
| TNP        | Pulse Circuit | 0.1 Å                | 2.710H           | 2.927H              | 163.6ns                  |
| SNP        | Pulse Circuit | 0.75 Å               | -1.137H          | -0.549H             | 71.1ns                   |
| TNP        | Pulse Circuit | 0.75 Å               | -1.137H          | -1.032H             | 163.6ns                  |
| TNP + SNP  | Pulse Circuit | 0.75 Å               | -1.137H          | -1.036H             | 234.7ns                  |
| Two Gate Ansatz | Gate Circuit | 0.75 Å               | -1.137H          | -0.534H             | 341.3ns                  |

occurrence of the barren plateau. In conclusion, frequency tuning can be considered as real-time calibration for pulses.

C. Ansatz Construction

Variational quantum algorithms usually consist of parametric circuits with a fixed structure. The main focus in the current VQA research community is to address an efficient way to find the configuration of ansatzes. Traditional ways of finding ansatzes are mostly physics or chemistry inspired, which may be less intuitive and cost-inefficient. For instance, to achieve certain accuracy for algorithms utilizing time-evolving blocks such as QAOA, a few trotter steps may be sufficient [80], and a shallow circuit can realize it. However, the structured ansatzes are not adaptive, and it may be wasteful when we aim to achieve a less demanding accuracy. Our progressive method generates pulse ansatz adaptively. The corresponding advantage is that the circuit depth is tailored for arbitrary desired accuracy and does not cause any experimental resource overhead. The next few sections illustrate the core steps in ansatz construction in detail.
TABLE III: Comparison of duration, pulse counts, and estimated energy of gate ansatz and the native-pulse ansatz generated by PAN on NISQ machines.

| Model               | Ansatz Level | Qubits | Duration | Single-Qubit Pulse Count | Multi-Qubit Pulse Count | Molecule | Energy | Reference Energy |
|---------------------|--------------|--------|----------|--------------------------|-------------------------|----------|--------|------------------|
| Random Genrated Ansatz | Gate Ansatz  | 2      | 682.7ns  | 16                       | 2                       | $H_2$    | -0.853 | -1.137           |
| RealAmplitude Ansatz [2] | Gate Ansatz  | 2      | 376.9ns  | 12                       | 1                       | $H_2$    | -0.974 | -1.137           |
| QuantumNAS [75] | Gate Ansatz  | 2      | 682.7ns  | 16                       | 2                       | $H_2$    | -1.033 | -1.137           |
| PAN                 | Pulse Ansatz | 2      | 71.1ns   | 3                        | 0                       | $H_2$    | -1.100 | -1.137           |
| RealAmplitude Ansatz | Gate Ansatz  | 2      | 753.8ns  | 24                       | 2                       | $HeH+    | -2.691 | -2.863           |
| PAN                 | Pulse Ansatz | 2      | 199.1ns  | 1                        | 1                       | $HeH+    | -2.866 | -2.863           |
| QuantumNAS | Gate Ansatz  | 6      | 7296.0ns | 40                       | 12                      | $LiH$    | -6.914 | -7.882           |
| PAN                 | Pulse Ansatz | 4      | 199.1ns  | 4                        | 2                       | $LiH$    | -7.590 | -7.882           |

Fig. 7: During the training phase, we are able to determine which part of the parametric pulse is trained. The complete list of parameters comprises a fixed list and a “partial” list. The parameters in the fixed list are not modified during the iteration. The parameters in the “partial” list are modified instead. This is how pulse ansatz is constructed in order to maximize the flexibility of choosing different policies. We also apply naive pruning approaches during training to further simplify the pulse ansatz.

D. Progressive Pulse Learning

Figure 6 gives an example of a policy to grow the native-pulse ansatz. This policy is inspired by the hardware-efficient ansatz (HEA) proposed by [43]. Similar to HEA, single-qubit native pulses are applied to all qubits, while two-qubit native pulses are applied to available connections between qubits. The difference between PAN and HEA is whether a progressive policy is adopted. The progressive manner of PAN can limit the number of parameters held by the optimizer to help with optimizer’s efficiency. Figure 7 gives more details on our progressive policy. During each new step, appended native pulses are trained, while previous pulses are kept and fixed. Such progressive manner resembles the idea of quantum speed limit, which denotes the maximum rate of evolution of a quantum system, i.e., the minimum time required for a quantum system to evolve between two quantum states [14], [60], [62]. Therefore, the first step of PAN might fail to reach a good enough quantum state. But it will gradually approximate the desired state with more steps.

Pruning. We propose to prune the pulse ansatz by removing pulses with parameters that are closest to zeros. The purpose of pruning is to reduce the overall pulse latency and decoherence error. Also, controls are less robust when their amplitudes are close to zeros. It is possible that pruning might force PAN to search in a sub-optimal space, but the progressive way can compensate for possible accuracy degradation. Experimental results show that pruning can significantly reduce the overall latency by an average of 24.13% on small molecule VQE tasks such as $H_2$, $HeH+$, and $LiH$, while accuracies experience minimal decreases.

Analysis of Effectiveness and Scalability of Optimizer. The pulse-level optimization algorithm has a drawback that we cannot use gradient based method like parameter-shift rule at the gate level. That is, we cannot obtain the gradients from quantum backends. Non-gradient optimizer becomes the only choice for us to use in the PAN framework. We use COBYLA in PAN, but the selection of non-gradient optimizer is an open problem. Non-gradient optimizers have two common drawbacks: they can only handle a limited number of parameters and they do not guarantee convergence to the global optimum. Therefore, we propose training the ansatz in a “progressive” manner, which helps to limit the number of parameters being trained at the same time.

To demonstrate how PAN mitigates the drawbacks of non-gradient optimizers, we conduct experiments in Fig. 8. In Fig. 8, there are three subplots, corresponding to three different settings for the same VQE task. “Ctrl-VQE” represents our implementation of Ctrl-VQE [54], where pulse segments are built and the parameters are trained all together. When “Ctrl-VQE” is proposed, it only considers single-qubit pulse channel which is not sufficient for VQAs. Therefore, we add two-qubit pulse channel for “Ctrl-VQE”, which can be regarded as an enhanced “Ctrl-VQE”. Proper considerations of two-qubit pulse channel also provide PAN with advantages over “Ctrl-VQE”. “PAN” represents the implementation of PAN, where pulse ansatizes are built and trained in a “progressive” way. In this manner, only a portion of parameters are trained simultaneously. “Brute-force pulse ansatz” can be viewed as a variant of PAN with only one “step” and a very large initial pulse, which means that all parameters are trained together. These experiments are conducted on simulators, each with 1024 shots for measurements. The blue lines stand for the mean of calculated molecule energies during training process.
And the green bands represent the variance of 20 different runs with different random seeds. We can tell from the figures that the proposed PAN converges faster than “Ctrl-VQE” and “Brute-force pulse ansatz”, while they share very similar accuracies. Thus, the progressive method is proven to be a good trade-off. Even though PAN leads to a sub-optimal result which is slightly worse than brute-force method, PAN has obvious advantages in convergence speed. Overall, PAN has greater potentials to hold middle-to-large size tasks.

As non-gradient optimizer is adopted in PAN, it is generally hard to illustrate how its overhead scales with the problem size. Since the performance of the non-gradient optimizer is mainly determined by the number of parameters, which is adaptive in our progressive learning framework. In PAN, we can decide the number of steps as well as the number of iterations in each step. Besides, we can choose the number of pulses that are appended in each step. Overall, the amount of parameters handled by the optimizer is very flexible. To demonstrate the convergence speed of non-gradient optimizer in PAN, we conduct experiments of training gate-ansatz and pulse-ansatz on the same task. Fig. 10 shows how estimated molecule energies change with the number of iterations. The results are obtained from real NISQ machines. Though PAN needs more iterations to converge than gate-based method, PAN actually needs less quantum resources. This is due to the fact that for gradient-based methods, parameter-shift rules need the circuits to be executed twice with shifted parameters to obtain gradients for each parameter. For a simple $H_2$ task, a two-parameter gate ansatz needs four runs of quantum circuits in one iteration. In Fig. 10, the gate-based ansatz uses 25 iterations to converge while pulse-ansatz requires 76. But in terms of overhead, the ratio is 100 to 76 rather than 25 to 76. For variational quantum algorithms, it is worth noting that gradient-based methods use parameter-shift rules to obtain gradients for ansatz parameters. Parameter-shift rules are similar to finite differences approximations of derivatives, where the “finite difference” is π. But instead of calculating approximations of derivatives, parameter-shift rules are returning exact gradients. In conclusion, for variational quantum algorithms, there is no major gap between performance of gradient-based methods and that of non-gradient methods.

VI. Evaluation

A. Backend configuration

Our experiments are conducted both on simulators and NISQ machines. Simulators are used to validate that PAN considers the system models of quantum backends. Simulations are run on a server with two Intel Xeon E5-2630 CPUs (8 cores/CPU), 64 GB DRAM, with CentOS 7.4 as the operating system. To confirm that our framework works on NISQ machines, we conduct experiments on six IBM’s quantum systems: ibm_cairo, ibmq_montreal, ibmq_toronto, ibmq_mumbai, ibmq_guadalupe and ibmq_jakarta. And we choose VQE problems as our evaluation tasks, which consist of several molecules including $H_2$, $HeH^+$ and $LiH$. During the progressive learning process, each step contains up to 50 optimization iterations, and each quantum program is executed 1024 shots for sampling.

B. Native pulse

In the Table II, we compare the results of different ansatz types for VQE tasks. The results are collected using NISQ machines. To determine the capabilities of our native-pulse model, we employ the simplest native-pulse model and compare its performance to that of native gates. The Table II demonstrates that our native-pulse approach can deliver superior VQE results in less durations. Pulse ansatz with only two native pulses can produce -1.036H energy, whereas the two-gate ansatz can only achieve -0.534H energy. The two-gate ansatz consists of both single-qubit and double-qubit gates. The duration of an ansatz ansatz is approximately 30% less than that of a gate ansatz. The duration numbers are calculated from the pulse schedules that produced by the same quantum backends. On simulators, we are able to produce superior VQE results with significantly less latency.

C. Simulation results

To confirm that our framework can generate accurate energy curves for given molecules, we use simulators to evaluate our methodologies. The specifications of “fake” backends are collected by the simulators and used as Hamiltonian configurations to execute the pulse simulation. We use pulse simulator that is provided by IBM qiskit toolkit. It simulates...
TABLE IV: Results of Estimated Energy for Molecules in Different Steps

| Model | Cairo | Montreal | Toronto | NISQ machine Avg | Simulator | FCI |
|-------|-------|----------|---------|------------------|-----------|-----|
| $H_2$ | Step I | -1.093 (3.870%) | -1.087 (4.398%) | -1.073 (5.629%) | -1.084 (4.661%) | -1.121 (1.407%) | -1.137 |
|       | Step II| -1.107 (2.639%) | -1.110 (2.375%) | -1.073 (5.629%) | -1.097 (3.518%) | -1.123 (1.231%) | -1.137 |
|       | Inaccuracy Reduction | 31.83% | 46.00% | 0.000% | 24.52% | 12.51% | - |
| $HeH^+$| Step I | -2.813 (1.746%) | -2.845 (0.663%) | -2.820 (1.485%) | -2.826 (1.292%) | -2.855 (0.279%) | -2.863 |
|       | Step II| -2.833 (1.047%) | -2.866 (0.105%) | -2.834 (1.013%) | -2.844 (0.664%) | -2.856 (0.244%) | -2.863 |
|       | Inaccuracy Reduction | 40.03% | 84.16% | 31.78% | 48.61% | 12.54% | - |

Fig. 9: a) Simulation results for $H_2$ molecule. The pulse ansatz curves mostly match the FCI value before Coulson-Fischer point [21]. The lowest energy point reaches accuracy of 97.69%. b) Simulation results of $HeH^+$ molecule. The pulse ansatz curve almost match the FCI value for all shown bond length. The lowest energy point reaches accuracy of 99.756%.

Fig. 10: Energy trends with #iterations for a $H_2$ VQE task. The data is collected from $ibmq_{\text{montreal}}$. a) Gradient-based method with parameter-shift rules for gate ansatz. b) PAN with non-gradient optimizer for pulse-level optimization. The vertical dotted line in the middle separates the steps in progressive learning. We can see several “peaks” on the curves because non-gradient optimizer attempts might be made in a bad “direction”.

results are only 0.244% off from the FCI results. In the case of $H_2$ molecules, our simulated results deviate from FCI data by 1.23%. As shown in Table IV, the insertion of a native pulse in STEP two improved the performance of the native-pulse ansatz by approximately 12% on simulators. We simulate gate circuits on noisy gate simulator and pulse schedules on noiseless pulse simulator. Despite the pulse simulator’s absence of noise, the algorithmic error is significant. It is unfair to compare the results from gate simulator and those from pulse simulator. Therefore, we present the results for gate-ansatz and pulse-ansatz on real NISQ machines in Fig. 10. Overall, the simulation experiments justify our native-pulse
Even with a narrow frequency shift range, we notice obvious changes in the estimated energies. The experimental results show that as we vary the detuning frequency, the energy estimation scans over a broad range, which validates our methods to tune frequency in pulses.

![Fig. 11: Influence of frequency shift on the estimated energies.](image)

**Fig. 11:** Influence of frequency shift on the estimated energies. Even with a narrow frequency shift range, we notice obvious changes in the estimated energies. The experimental results show that as we vary the detuning frequency, the energy estimation scans over a broad range, which validates our methods to tune frequency in pulses.

![Fig. 12: Evaluation of native-pulse ansatz on NISQ machines for $H_2$, HeH+, and LiH VQE tasks.](image)

**Fig. 12:** Evaluation of native-pulse ansatz on NISQ machines for $H_2$, HeH+, and LiH VQE tasks. NISQ machines include ibm_cairo, ibmq_montreal, ibmq_toronto, ibmq_mumbai, ibmq_guadalupe, and ibmq_jakarta. Our toy ansatzes are able to achieve great accuracies on all the NISQ machines. For ibmq_montreal where we obtain the best results, the average CNOT error is 1.518% as the average readout error is 3.457%. PAN is proven to be robust and error-resilient on NISQ machines.

![Fig. 13: Evaluation of native-pulse ansatz on NISQ machines for $CO_2$, $H_2O$, and NaH’s VQE tasks as well as three-regular six nodes graph QAOA task.](image)

**Fig. 13:** Evaluation of native-pulse ansatz on NISQ machines for $CO_2$, $H_2O$, and NaH’s VQE tasks as well as three-regular six nodes graph QAOA task. NISQ machines include ibmq_montreal and ibmq_jakarta.

D. NISQ machine results

Our techniques are also evaluated on several NISQ computers. As presented in Figure 12 and Table III, a toy model of native-pulse ansatz is capable of producing promising results. In the instance of the HeH+ molecule, we attain an average accuracy of 99.336%, with 99.895% being the highest achievable accuracy. The absolute difference in energy is 0.003H. The accuracy is very close to the requirement of computational chemistry (0.0016H), which is the chemical accuracy constant for computational chemistry. It qualifies the typical minimum energy gap that can be verified through experiments. As for the $H_2$ molecule, we attain an average accuracy of 96.482% and a maximum accuracy of 97.625% using the same native-pulse ansatz. For LiH, a bigger molecule than those in the preceding cases, we obtain an energy accuracy of 96.295%. These accuracy figures are derived from NISQ machines with gate and measurement errors exceeding 1%. We observe that ibmq_montreal tends to return the best results, while ibmq_mumbai typically returns the worse results. Then, we confirm that ibmq_mumbai has greater error rates for gates and measurements on average than ibmq_montreal.

The results collected from NISQ computers demonstrate that our approaches are highly error-tolerant. In terms of total duration time, our native-pulse approaches have significant advantages over the current gate ansatz generator. To ensure equality, the gate ansatz of the baselines is implemented on the same NISQ computer, and their duration is determined using the acquired pulse schedules. We are able to demonstrate
a 97.3% reduction in ansatz duration compared to QuantumNAS, and our estimated energy numbers for LiH are lower. With a duration decrease of 89.6%, our native-pulse method can obtain better energy values while dealing with H2 molecules. QuantumNAS does not report the energy numbers for the HeH+ molecule. Thus we compare our techniques to the Real Amplitude Ansatz, which shows that we are able to reduce duration by 73.6% while maintaining similar performance.

Figure 10b depicts the relationships between the computed energy and the number of iterations. Despite the fact that it is not visible in Figure 10b, we validate that our progressive approaches work on VQE tasks. Since a non-gradient optimizer is deployed, we can observe several peaks of the curve where the non-gradient optimizer attempts to update the pulse ansatz’s parameters but obtains poorer results. As indicated in the Table IV, we are able to reduce the deviation from FCI values by around 40%, when native-pulses “grow” progressively. Only in one instance where the pulse ansatz were conducted on ibmq_toronto were the results not improved. Taking into account this failure, we find a 24% improvement for H2 molecule experiments and a 48% improvement for HeH+ molecule experiments.

We also evaluate PAN on larger VQE tasks as well as QAOA task as show in Fig. 13. For CO$_2$, H$_2$O, and NaH, the pulse ansatz generated by PAN has a duration of 1031.1 ns on ibmq_montreal and the estimated ground state energies are -183.856H (99.70% accuracy), -73.574H (97.96% accuracy), and -150.984H (94.15% accuracy), respectively. For three-regular six node graph, experiment is conducted on NISQ machine ibmq_jakarta with resulting improved approximation rates of 3.4% compared to the same duration gate ansatz. Results demonstrate feasibility of PAN on different VQAs. Here is one more thing need to noticed, since the hamiltonian layer in QAOA question is a discrete mapping and fixed with ‘RZZ’ structure that hard to do progressive in Hamiltonian layer, thus we using the progressive learning only on mixer layer and gain 40% duration reduction on mixer layer. A theoretical designing of the problem encoding on Hamiltonian layer should be able to further improve the function of PAN.

E. Tuning frequencies

In Fig. 14, we detune the pulse frequency and use a few steps to estimate the energy of the H2 molecule. The experimental results show that as we vary the detuning, the energy estimation scans over a broad range, which ensures that the pulse parameter variation in the learning step is wide enough to cover the ground energy. Moreover, the evident response of energy estimation with pulse parameterization shows that our ansatz structures are valid in covering all spin-orbital configurations of the electronic structure. The estimated energy can then progressively approximate the desired ground energy through pulse parameters learning.

VII. Related Work

Pulse Learning Approach: Ctrl-VQE [54] changes the pulse shape to perform state preparation. The methods are evaluated with QuTiP [42] pulse simulator. And the results demonstrate that the coherence time required for state preparation is greatly reduced. VQP [49] uses pulses as basic components to build the QNN ansatz and exhibits latency advantages over gate-based QNN on a two-class image classification task. The experiments are mostly on Qiskit [52] pulse simulator. VQOC [25] presents a mathematical formalism of optimal control, which acts on pulse optimization for VQA tasks. Their method is similar to Ctrl-VQE [54], but they take advantages of neutral atom’s properties. The evaluations are also performed on simulator. These previous works are exploratory and rely on classical simulations of small quantum systems [19], [25], [49], [54]. PAN, on the other hand, provides results from NISQ machines. Moreover, Ctrl-VQE is only designed the consideration for single qubit pulse, whereas, PAN both consider single qubit pulse and two qubit pulse.

Ansatz Architecture Search: Hamiltonian simulation plays an important role in simulating quantum systems [43], [43], [59], [70]. Variational quantum algorithms can be deployed to perform Hamiltonian simulation. And previous works [43] pointed out that the choice of ansatz is important for VQAs. The conventional approaches of choosing ansatz depends heavily on the applications. [10], [57], [59] are specially designed for VQE tasks. The unitary coupled-cluster singles and doubles schele (UCCSD) is still the golden standards for VQE ansatz. UCCSD approximates the states of the molecule and can be implemented with qubits. QAS [28] proposes a noise-aware scheme to search for ansatz structure. The robustness to noise is demonstrated on simulators. QuantumNAS [75] presents a comprehensive framework for noise-adaptive co-search of the ansatz. Hardware topology is considered during the search algorithm. QuantumNAS validates their methods with evaluations on NISQ machines.

Pulse Gate Compilation: [34] proposes a new compilation paradigm, based on the OpenPulse interface for IBM quantum computers. This work achieves lower error rates and shorter execution times in comparison with traditional gate-based compilation methods. Their technique is bootstrapped from existing gate calibrations, thus their pulses are in a simple form. [64] designs a general quantum compilation method to integrate multiple operations into larger units and then achieve high efficiency by optimizing this aggregate and creating corresponding custom control pulses.

VIII. Conclusion

PAN is a framework of constructing native-pulse anstaz for variational quantum algorithms. As a result of removing an abstraction layer of native gates, native pulses provide huge latency advantages. Then, we employ progressive learning to “grow” our ansatz. Thus, our pulse ansatz is better able to explore the Hilbert space, while the optimizer is still able to handle the problem. Extensive experiments are conducted on NISQ machines, and the results indicate that an average of
86% reduction in latency is obtained with up to 99.895% accuracy on small molecule VQE task. Experiments on larger-size molecule VQE tasks achieve an average accuracy of 97.27%. The results of PAN on QAOA demonstrate the feasibility of PAN on different VQAs. How to explore the applications of PAN on different types of VQAs remains an open question. Also, potentials of PAN on quantum algorithms other than VQAs are to be investigated. Overall, our experimental results show that pulse-level methods like PAN can greatly reduce the duration of quantum circuits, thus enhance the capabilities of quantum computers.
[42] J. R. Johansson, P. D. Nation, and F. Nori, “Qutip: An open-source python framework for the dynamics of open quantum systems,” Computer Physics Communications, 2012.

[43] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” Nature, vol. 540, no. 7611, pp. 242–246, 2017.

[44] J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Y. Chen et al., “State preservation by repetitive error detection in a superconducting quantum circuit,” Nature, vol. 519, no. 7541, pp. 66–69, 2015.

[45] P. Kok, W. J. Munro, K. Nemoto, T. C. Ralph, J. P. Dowling, and G. J. Milburn, “Linear optical quantum computing with photonic qubits,” Reviews of modern physics, vol. 79, no. 1, p. 135, 2007.

[46] N. Leung, M. Abdelhafez, J. Koch, and D. Schuster, “Speedup for quantum optimal control from automatic differentiation based on graphics processing units,” Physical Review A, vol. 95, no. 4, p. 042318, 2017.

[47] G. Li, Y. Shi, and A. Javadi-Abhari, “Software-hardware co-optimization for computational chemistry on superconducting quantum processors,” in 2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA). IEEE, 2021, pp. 832–845.

[48] G. Li, A. Wu, Y. Shi, A. Javadi-Abhari, Y. Ding, and Y. Xie, “Paulithedral: a generalized block-wise compiler optimization framework for quantum simulation kernels,” in Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2022, pp. 554–569.

[49] Z. Liang, H. Wang, J. Cheng, Y. Ding, H. Ren, X. Qian, S. Han, W. Jiang, and Y. Shi, “Variational quantum pulse learning,” arXiv preprint arXiv:2203.17267, 2022.

[50] J.-G. Liu, Y.-H. Zhang, Y. Wan, and L. Wang, “Variational quantum eigensolver with fewer qubits,” Physical Review Research, vol. 1, no. 2, p. 023025, 2019.

[51] M. Lusbach, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, “Variational quantum algorithms for nonlinear problems,” Physical Review A, vol. 101, no. 1, p. 010301, 2020.

[52] D. C. McKay, T. Alexander, L. Bello, M. J. Biercuk, L. Bishop, J. Chen, J. M. Chow, A. D. Córcoles, D. Egger, S. Filipp et al., “Qiskit backend specifications for openqasm and openpulse experiments,” arXiv preprint arXiv:1809.03452, 2018.

[53] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, “Efficient z gates for quantum computing,” Phys. Rev. A, vol. 96, p. 022330, Aug 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.96.022330

[54] O. R. Meitei, B. T. Gaed, G. S. Barron, D. P. Pappas, S. E. Economou, E. Barnes, and N. J. Mayhall, “Gate-free state preparation for fast variational quantum eigensolver simulations,” npj Quantum Information, vol. 7, no. 1, pp. 1–11, 2021.

[55] N. Moll, P. Barkoutos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn et al., “Quantum optimization using variational algorithms on near-term quantum devices,” Quantum Science and Technology, vol. 3, no. 3, p. 030503, 2018.

[56] P. Neumann, N. Mizuochi, F. Rempp, P. Hemmer, H. Watanabe, S. Yamasaki, V. Jacques, T. Gaebel, F. Jelezko, and J. Wrachtrup, “Variational quantum eigensolver simulations,” EPL (Europhysics Letters), vol. 123, no. 6, p. 60001, Oct 2018. [Online]. Available: https://doi.org/10.1209%2F0295-5075%2F123%2F60001

[57] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth et al., “The variational quantum eigensolver: a review of methods and best practices,” arXiv preprint arXiv:2111.05176, 2021.

[58] T. Tomesh, P. Gokhale, V. Omole, G. S. Ravi, K. N. Smith, J. Vizelai, X.-C. Wu, N. Hardavellas, M. R. Martinoson, and F. T. Chong, “Supermarq: A scalable quantum benchmark suite,” in 2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA). IEEE, 2022, pp. 587–603.

[59] T. Tomesh, K. Gui, P. Gokhale, Y. Shi, F. T. Chong, M. Martinoson, and M. Suchara, “Optimized quantum program execution ordering to mitigate errors in simulations of quantum systems,” in 2021 International Conference on Rebooting Computing (ICRC). IEEE, 2021, pp. 1–13.

[60] L. M. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, “ Experimental realization of shor’s quantum factoring algorithm using nuclear magnetic resonance,” Nature, vol. 414, no. 6866, pp. 883–887, 2001.

[61] D. Wang, O. Higgott, and S. Brierley, “Accelerated variational quantum eigensolver,” Physical review letters, vol. 122, no. 14, p. 140504, 2019.

[62] H. Wang, Y. Ding, J. Gu, Y. Lin, D. Z. Pan, F. T. Chong, and S. Han, “Quantumins: Noise-adaptive search for robust quantum circuits,” in 2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA). IEEE, 2022, pp. 692–708.

[63] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, “Noise-induced barren plateaus in variational quantum algorithms,” Nature Communications, vol. 12, no. 1, pp. 1–11, 2021.

[64] C. Weedbrook, S. Pirandola, R. García-Patrón, N. J. Cerf, T. C. Ralph, J. H. Shapiro, and S. Lloyd, “Gaussian quantum information,” Rev. Mod. Phys., vol. 84, p. 621–669, May 2012. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.84.621

[65] J. Wierschink and E. Gross, “Quantum optimal control theory,” Journal of Physics B: Atomic, Molecular and Optical Physics, vol. 40, no. 18, p. R175, 2007.

[66] X. Yang, X. Nie, Y. Ji, T. Xin, D. Lu, and J. Li, “Improved quantum computing with the higher-order trotter decomposition,” 2022. [Online]. Available: https://arxiv.org/abs/2205.02520