Hierarchical Multigrid Ansatz for Variational Quantum Algorithms

Christo Meriwether Keller*†, Stephan Eidenbenz†, Andreas Bärtschi†,
Daniel O'Malley‡, John Golden†, Satyajayant Misra*
*Department of Computer Science, New Mexico State University, Las Cruces NM 88003, USA
†Information Sciences (CCS-3), Los Alamos National Laboratory, Los Alamos NM 87544, USA
‡Computational Earth Sciences (EES-16), Los Alamos National Laboratory, Los Alamos NM 87544, USA

***Abstract*—Quantum computing is an emerging topic in engineering that promises to enhance supercomputing using fundamental physics. In the near term, the best candidate algorithms for achieving this advantage are variational quantum algorithms (VQAs). We design and numerically evaluate a novel ansatz for VQAs, focusing in particular on the variational quantum eigensolver (VQE). As our ansatz is inspired by classical multigrid hierarchy methods, we call it "multigrid" ansatz. The multigrid ansatz creates a parameterized quantum circuit for a quantum problem on n qubits by successively building and optimizing circuits for smaller qubit counts $j < n$, reusing optimized parameter values as initial solutions to next level hierarchy at $j+1$. We show through numerical simulation that the multigrid ansatz outperforms the standard hardware-efficient ansatz in terms of solution quality for the Laplacian eigensolver as well as for a large class of combinatorial optimization problems with specific examples for MaxCut and Maximum k-Satisfiability. Our studies establish the multi-grid ansatz as a viable method for improving the performance of variational quantum eigensolvers.**

***Index Terms*—Variational quantum algorithms, multigrid methods, discrete Laplacian operators**

I. Introduction

Variational quantum algorithms are hybrid methods for gate-based quantum computers that combine quantum and classical techniques. By partially offloading error-prone quantum processing to classical algorithms, these methods present the best opportunity for achieving quantum advantage in the Noisy Intermediate Scale Quantum (NISQ) era. Due to this potential for impact, researchers continue to look for improvements to variational quantum algorithms.

The VQE algorithm was proposed first by Peruzzo et al. [1]. Following on this, a variational quantum algorithm for solving the Poisson equation was given by Sato et al. [2]. This work was further developed on by Liu et al. [3] who improved the number of measurements required from exponential to linear (making it practical) and connected it with QAOA. A comprehensive review of the VQE space is given by Tilley et al. [4], and a gentle introduction by Adedoyin et al. [5].

Other variants of VQE have been proposed, such as ADAPT-VQE [6]. This algorithm iteratively produces larger ansätze by Trotterization to solve chemical simulation problems. Our work differs from this in that we aim for problems where there is a natural refinement structure, and especially **NP**-hard optimization problems such as MaxCut. In addition, our approach generalizes a simple case of classical mesh refinement, where setting new parameters to zero corresponds to the output of the previous refinement layer. Another variant in this vein of "growing ansätze" is Layer VQE [7], which increases the depth of its ansatz at each stage. This method grows each successive ansatz in a completely "different direction" than our work does, but is applied to combinatorial optimization problems.

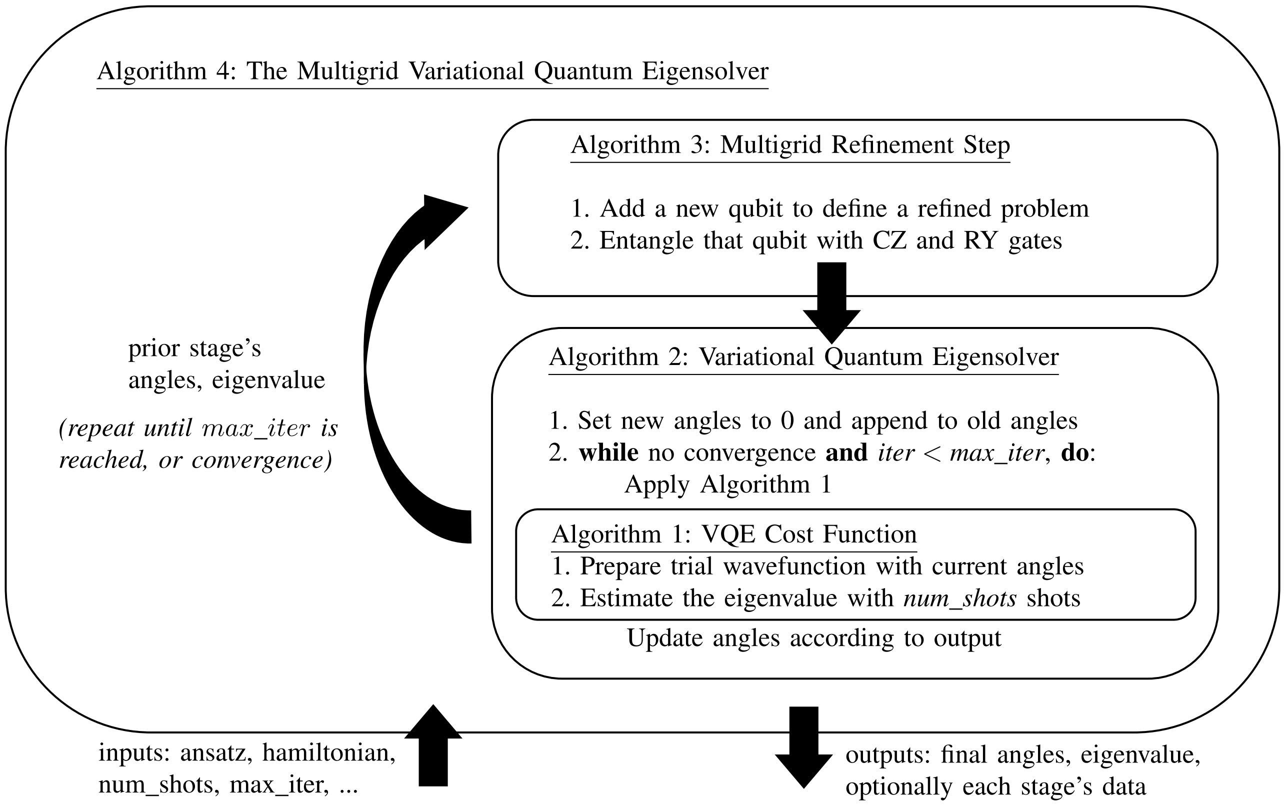
In this paper, we introduce multigrid methods for the variational quantum eigensolver (VQE) and test them against other quantum approaches and classical algorithms. Multigrid methods use hierarchies of discretizations to solve problems (and especially differential equations) by moving from coarser to finer grids. This process, known as mesh refinement, is the algorithm for which we propose a variational quantum counterpart. Also, there is much potential for future work in this area: for example, more sophisticated multigrids and connections with other algorithms such as variational linear systems [8].

Multigrid methods have been applied to quantum annealers by Illa and Savage [9] and Lubasch et al. adapted finite-difference methods to a VQA for nonlinear PDEs [10]. However, we are not aware of any published work developing multi-grid methods (in particular, mesh refinement) for gate-based quantum computing architectures. We propose methods for platforms with all-to-all qubit connectivity, opening the pathway for future work to optimize these for other topologies.

In Section II, we introduce variational quantum algorithms more thoroughly and explain two important examples. We also discuss hardware-efficient quantum ansätze, which are used as a benchmark later on. Section III discusses our methods, including Multigrid Hierarchies of Variational Ansätze and the Multigrid VQE. In Section IV, we discuss how to implement our methods for discrete Laplacian energy simulation, and our results for that problem. Then, in Section V, we look at **NP**-hard Maximum Cut problem on graphs. Section VI concludes with a brief recap of the paper, and directions for future work.

II. Background

For a general overview of quantum optimization, we recommend references [11] and [12]. The rest of this section will focus on variational quantum algorithms, and particularly our



Algorithm 4: The Multigrid Variational Quantum Eigensolver
Algorithm 3: Multigrid Refinement Step
1. Add a new qubit to define a refined problem
2. Entangle that qubit with CZ and RY gates
Algorithm 2: Variational Quantum Eigensolver
1. Set new angles to 0 and append to old angles
2. while no convergence and iter < max_iter, do:
Apply Algorithm 1
Algorithm 1: VQE Cost Function
1. Prepare trial wavefunction with current angles
2. Estimate the eigenvalue with num_shots shots
Update angles according to output
prior stage's angles, eigenvalue
(repeat until max_iter is reached, or convergence)
inputs: ansatz, hamiltonian, num_shots, max_iter, ...
outputs: final angles, eigenvalue, optionally each stage's data

Fig. 1: A process diagram for the Multigrid VQE. When one level of the Hamiltonian is optimized, refine the problem and use the previous level's angles to optimize. This process procedes until refinement produces the original Hamiltonian.

description of the variational quantum eigensolver.

A. Variational Quantum Algorithms

Variational quantum algorithms (VQAs) are an active area of research in quantum information science, with the potential for achieving quantum advantage on near-term quantum computers [13]. They center on a parameterized quantum circuit $U(\theta_1, \theta_2, \ldots, \theta_n)$ called an *ansatz* which prepares a trial wavefunction. This quantum state is measured in various ways, and those measurements are used to classically calculate a score. This score is fed to a classical optimization algorithm to update the parameters, and this optimization loop proceeds until convergence or a maximum number of iterations has been reached. At this point the angles are finalized and the corresponding state can be prepared at will and used for whatever purpose is desired, or the final score can be returned as output.

Important VQAs include the Variational Quantum Eigensolver (VQE) for finding the ground state of a Hamiltonian [1], and the Quantum Approximate Optimization Algorithm (QAOA) for combinatorial optimization problems [14]. In the long term, VQE has the potential to find applications to chemistry, and QAOA to become a serious competitor for real-world combinatorial optimization applications. As a result, considerable effort has been spent developing methods for these VQAs, and studying how different approaches affect performance across different classes of problems. For example, ore sophisticated variants have been proposed, such as filtered VQE, warm-start QAOA, and recursive QAOA [15]–[17].

These hybrid approaches allow for shallower and more robust circuits, offloading quantum complexity into more manageable classical overhead. However, such methods come with substantial uncertainty; no speedup has been proven for VQAs, and their cost landscapes are susceptible to *barren plateaus* [18] where classical optimizers get caught descending into infinitely vanishing (or, in the case of maximization, infinitely growing) regions. However, preliminary experimental work by Boulebnane and Montenaro [19] and Golden et al. [20] suggest the potential for QAOA advantage. Prior work by Farhi et al. [21] on Max-kXOR had achieved a better guaranteed bound than the best known classical methods at the time, but better classical methods were subsequently found by Barak et al. [22].

B. Relevant VQA Methods

1) Variational Quantum Eigensolver: Given a Hamiltonian

$$H = \sum_i W_i^\dagger \alpha_i P_i W_i, \tag{1}$$

represented as a sum of unitary-transformed weighted Pauli strings (as in Equation (1), where W_i are unitary and α_i are weights), prepare a trial state $U(\theta_1, \theta_2, \ldots, \theta_n)$ using a VQA ansatz and calculate the weighted sum of measurements M_i of each Pauli string P_i on the corresponding

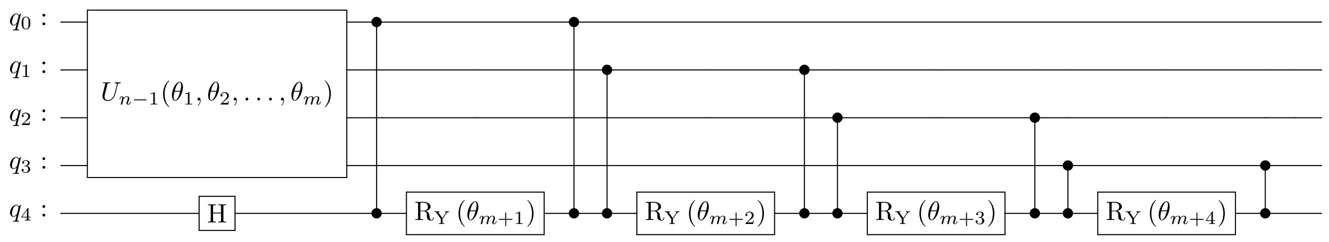

q_0 :
q_1 :
q_2 :
q_3 :
q_4 :
$U_{n-1}(\theta_1, \theta_2, \ldots, \theta_m)$
H
$\mathrm{R_Y}\ (\theta_{m+1})$
$\mathrm{R_Y}\ (\theta_{m+2})$
$\mathrm{R_Y}\ (\theta_{m+3})$
$\mathrm{R_Y}\ (\theta_{m+4})$

Fig. 2: A refinement layer of a Multigrid Hierarchy of Variational Ansätze. The circuit $U_{n-1}(\theta_1, \theta_2, \ldots, \theta_m)$ is the previous refinement layer with angles from the previous optimization round. The solution to the previous round's coarser problem on four qubits is extended to a refined problem on five qubits by putting a new qubit in the $|+\rangle$ state with an `H` gate and entangling it to all previous qubits with `CZ` (connected black dots) and `RY` gates. In our implementation, the new angles $\theta_{m+1}, \theta_{m+2}, \theta_{m+3}$, and θ_{m+4} are each initialized to zero in order to create a constant interpolation with respect to computational basis states.

state $W_i U(\theta_1, \theta_2, \ldots, \theta_n)$. Repeating this procedure many times and averaging yields an approximation of the Rayleigh quotient $\langle U(\theta)| H |U(\theta)\rangle / \langle U(\theta)|U(\theta)\rangle$, which is an upper bound on the Hamiltonian ground state energy $\lambda_0(H)$. (Note the denominator of this quotient is immaterial since quantum states have unit norm, and averaging works because the numerator is an expectation value.) Given an ansatz which can find this state and an effective optimization strategy, variationally minimizing this term can bring us arbitrarily close to (given enough shots) or even saturate this bound. Note that everything in this section also applies to Hamiltonians defined as weighted sums of Pauli strings without unitary transformation, by setting W_i to the corresponding identity matrix for each term in Eq. 1.

The VQE algorithm works like this, and can be seen as a quantum analogue of the classical Ritz method [23]. The absolute error scales down as the number of shots increases; heuristically, $\mathcal{O}(1/\varepsilon^2)$ shots are required [1] to obtain error within ε. However, this result relies on various assumptions about the robustness of the ansatz and the well-suitedness of the classical optimization routine for the energy landscape. Algorithm 1 gives the VQE cost function computing the Rayleigh quotient for a given set of angles. It does this by measuring each term in the decomposition of the target Hamiltonian many times, and averaging the measurement results. Algorithm 2 uses this function in the overall logic of the VQE algorithm, by varying and optimizingthe ansatz parameters until either a maximum number of iterations is reached or convergence is achieved. Figure 1 shows our modified VQE, involving also Algorithms 3 and 4 (defined in Section III). In order to measure arbitrary Pauli strings, this involves applying change of basis gates: apply H to a qubit to measure in the X basis, and apply $S^\dagger H$ to measure in the Y basis. Of course, decomposing H according to Equation (1) can be nontrivial, as can the efficient implementation of arbitrary unitaries W_i. In particular, representing an n-qubit Hamiltonian H as a sum of Pauli strings takes 4^n terms in general. Thus it is important to measure in more convenient bases. For example, Liu et al. [3] use the operator basis $\mathcal{B} = \{I, \sigma_+, \sigma_-\}$, where $\sigma_+ = |0\rangle\langle 1|$ and $\sigma_- = |1\rangle\langle 0|$, in their VQA for solving the Poisson equation.

2) Quantum Approximate Optimization Algorithm: Given a cost function C on bitstrings, we can encode the corresponding optimization problem in a diagonal matrix

$$H_C = \sum_{i \in 2^n} C(i)\mathbf{e}_i, \tag{2}$$

and approximately optimize C using a discretization and Trotterization of the Quantum Adiabatic Algorithm [14], [24]. This method is called QAOA and was first applied to the NP-complete Maximum Cut problem (MaxCut) for the maximum number of edges between two complementary vertex subsets of a given graph G. It is worth noting that this diagonal Hamil-

Algorithm 1 The VQE Cost Function (vqe_cost)

```
input: Ansatz $U$, Hamiltonian $H$, Angles $\theta_1, \theta_2, \ldots, \theta_m$, Number of shots $num\_shots$.
output: Estimated eigenvalue $\lambda$.

estimated_eigenvalue = 0;
$W_i$, $P_i$, $\alpha_i$ = Decompose Hamiltonian according to Eq. 1;
for $0 \le i < n$, do
    term_eigenvalue = 0;
    // $H.num\_qubits$ is the number of qubits in the system $H$
    circ = new QuantumCircuit($H.num\_qubits$)
    // Appending Q1 to Q2 means performing Q2 then Q1
    Append $U(\theta_1, \theta_2, \ldots, \theta_m)$ to circ;
    Append $W_i$ to circ;
    Add measurement gates for $P_i$ to circ;
    do $num\_shots$ times
        measure = Execute circ on quantum computer;
        test_eigenvalue = pow(−1, hamming(measure));
        term_eigenvalue += $\alpha_i$ ∗ test_eigenvalue/num_shots;
    end do
    estimated_eigenvalue += term_eigenvalue;
end for
return estimated_eigenvalue;
```

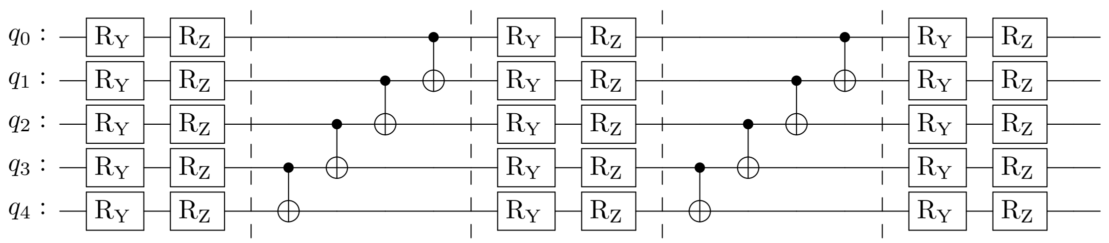


Fig. 3: An Efficient SU(2) ansatz with two repetitions and a final rotation layer. This circuit exhibits the *reverse linear* entanglement pattern, which is the default in Qiskit. Ladders of reverse linear entangling CX gates separate blocks of parameterized Y and Z rotations to form a simple yet robust ansatz for heuristic VQA applications.

tonian H_C can be fed into VQE if a suitable representation is known (see Section V for an example).

3) Hardware-Efficient Ansätze: In order to avoid excessive noise in a VQA, efficient operations and low circuit depth are desired. Hardware-efficient ansätze use native gates (such as nearest-neighbor interactions for superconducting qubits) to form a robust constant-depth circuit for preparing trial states [25]. These have been shown to suffer from barren plateaus at long depths, but to have the potential for quantum advantage when short [26]. Hardware efficient ansätze provide reasonable benchmarks for testing VQAs across different classes of problems due to their general nature and resultingly broad applicability.

III. Multigrid VQE

We employ an approach based on classical multigrid methods, where hierarchies of increasingly finer discretizations are used to solve problems, such as differential equations. First solving a coarser problem with an initial variational ansatz $U(\theta_1, \theta_2, \dots \theta_n)$, we save the optimized angles θ_i and use these to seed the next round. In this subsequent round, we apply the ansatz U (which we call the *seed ansatz*) then entangle it to a new qubit as shown in Figure 2. This method preserves the multigrid intuition because setting the initial angles for the RY gates entangling the new qubit to zero puts the system in the state $U(\theta_1, \theta_2, \dots, \theta_n)\left|00\dots0\right\rangle \otimes \left|+\right\rangle$, which corresponds to a constant interpolation of the state from the previous round. That is, if $U(\theta_1, \theta_2, \dots, \theta_n)\left|00\right\rangle = \left|\psi\right\rangle$, then

$$U(\theta_1, \theta_2, \dots, \theta_n)\left|00\right\rangle \otimes \left|+\right\rangle = \frac{\left|\psi\right\rangle\left|0\right\rangle + \left|\psi\right\rangle\left|1\right\rangle}{\sqrt{2}}, \quad (3)$$

the equal superposition of adding a $\left|0\right\rangle$ qubit and adding a $\left|1\right\rangle$ qubit. This occurs since neutralizing the Y rotations on the added qubit causes all CZ gates to annihilate, yielding simply a Hadamard operation returning the plus state on that qubit. We continue adding qubits until we reach the target system.

The two-qubit gate cost of this method is only quadratic in the number of qubits, keeping it reasonable for use on NISQ devices. In state preparation problems, this makes the gate cost polylogarithmic in the number of amplitudes (assuming polynomial circuits for any unitary transforms in the Pauli decomposition of the problem Hamiltonian, Equation (1)). This will be relevant in the next section, where we use Multigrid VQE to prepare the ground state of a discrete Laplacian Hamiltonian. The overall idea is that for problems with a hierarchical structure, information from easier, lower stages can be used to improve the performance of harder, higher stages. There is no particular reason this process has to proceed one qubit at a time, either; future work could explore

Algorithm 2 The VQE Classical Loop (vqe)

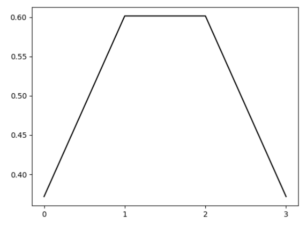
input: Ansatz U, Hamiltonian H, Angles $\theta_1, \theta_2, \dots, \theta_m$, Numbers max_iter, num_shots.
output: Estimated eigenvalue λ, Final angles $\boldsymbol{\theta}'_i$, Optional optimizer information (e.g. number of function evaluations).

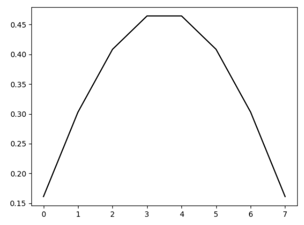
```
$num\_iter = 0$;
$estimated\_eigenvalue$ = vqe_cost($U, H, \boldsymbol{\theta}_i, num\_shots$)
while no convergence and $num\_iter < max\_iter$, do
    $\theta_1, \theta_2, \dots, \theta_m$ = optimize_angles($estimated\_eigenvalue$,
        $\theta_1, \theta_2, \dots, \theta_m$); // Using a classical optimizer.
    $estimated\_eigenvalue$ = vqe_cost($U, H, \theta_i, num\_shots$);
    $num\_iter$ += 1;
end while
return $estimated\_eigenvalue$, $\boldsymbol{\theta}_i$, optional info (such as number of optimizer calls or other classical optimizer results);
```

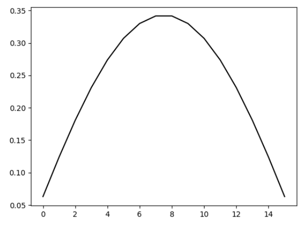
Algorithm 3 The Multigrid VQE (multigrid_vqe)

input: Seed ansatz U, Family of Hamiltonians $\{H_i\}$, Initial angles $\boldsymbol{\theta}_i$, Numbers num_shots, max_iter.
output: Estimated eigenvalue λ, Final angles $\boldsymbol{\theta}'_i$, Optional optimizer information and coarse stage data.

```
$\lambda, \theta_i$ = vqe($U, H, \theta_1, \theta_2, \dots, \theta_m, max\_iter, num\_shots$);
for $U.num\_qubits < j <= H.num\_qubits$, do
    $\theta_{m+1}, \theta_{m+2}, \dots, \theta_{m+U.num\_qubits} = 0, 0, \dots, 0$;
    $U_j$ = multigrid_refine($U_{j-1}$); // With $U_{U.num\_qubits} = U$
$\lambda, \theta_i$ = vqe($U_j, H_j, \theta_1, \theta_2, \dots, \theta_m, \dots, \theta_{m+U.num\_qubits}$,
        $max\_iter, num\_shots$);
    $m = m + U.num\_qubits$; // For the next iteration.
end for
return $\lambda, \boldsymbol{\theta}_i$;
```


(a) two qubits


(b) three qubits


(c) four qubits

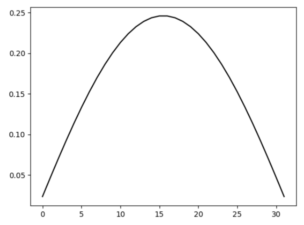

(d) five qubits

Fig. 4: The smallest eigenvectors of ∇_D^2 (for dimensions $m = 2^2, 2^3, 2^4, 2^5$, respectively) form a refinement structure well-suited to multigrid methods. The X axes represent computational basis states expressed as decimal integers, and the Y axes show the amplitudes of the eigenvector in the respective entries. Ignoring a complex global phase, we assume entries in $\mathbb{R}_+$.

Algorithm 4 Multigrid Refinement (multigrid_refine)

```
input: A parameterized quantum circuit $U_n$ with $n$ qubits.
output: A parameterized quantum circuit $U_{n+1}$ with $n+1$ qubits.

circ = new QuantumCircuit($U.num\_qubits + 1$);
Append $U$ to the first $U.num\_qubits$ qubits of circ;

// We assume that qubit addresses start at zero.
Append a Hadamard gate to qubit $n$ of circ;

for $i < n$, do
    Append a $CZ$ gate between qubits $i$ and $n$ of circ;
    Add a new variational parameter $p_{m+i}$ to circ;
    Append the gate $RY(p_{m+i})$ to qubit $n$ of circ;
    Append a $CZ$ gate between qubits $i$ and $n$ of circ;
end do
return circ;
```

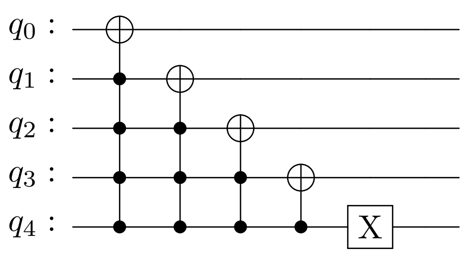


Fig. 5: An increment circuit for the shift operation P of Sato et al. [2]. This circuit sends big-endian basis states $|b\rangle$ to their cyclically shifted counterparts $|\mathtt{rem}(b+1, 2^n)\rangle$, and can be decomposed into linearly many CNOTs [27].

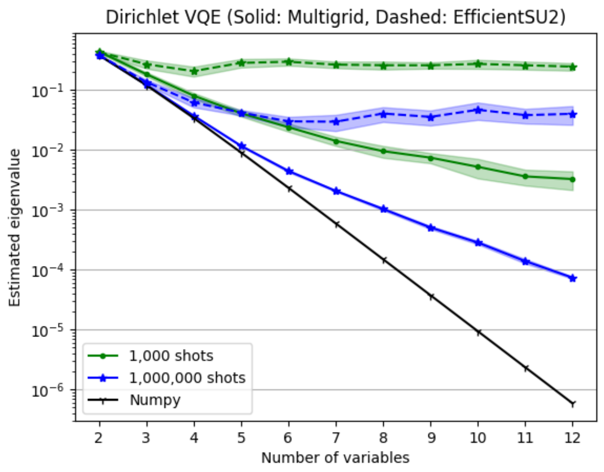
the integration of multiple new qubits at once, or designs that use fewer entangling gates in each consecutive layer for implementations on qubit topologies, like with IBM's heavy-hex lattice architecture, that lack all-to-all connectivity.

Algorithm 3 uses multigrid refinement to perform what we call multigrid VQE, assuming a hierarchy of properly represented Hamiltonians. The idea is to solve progressively larger Hamiltonians using information about each previous stage's angles until you reach your target Hamiltonian. Based on the design of the multigrid refinement function, we can think of this as an analogue of classical mesh refinement with constant interpolation. One drawback of this approach is that the maximum number of total optimizer iterations goes to infinity as the size of the problem scales up. This can be addressed by choosing larger seed ansätze for larger problems. In this work, we do not simulate large enough n for this to become an issue, so we use the Efficient SU(2) seed ansatz to standardize our experiments. Given an ansatz with a total of $n_{s.a.}$ parameters, the Multigrid Hierarchy has

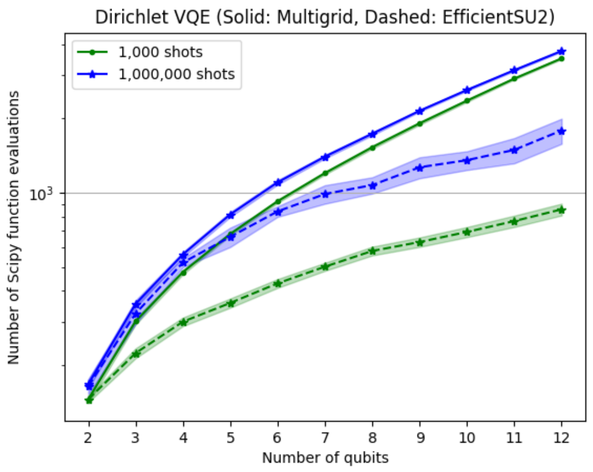
$$n_{s.a.} + \sum_{i=m}^{n-1} i = n_{s.a.} + \frac{n^2 - n - m^2 + m}{2} \tag{4}$$

parameters. Now $m = 2$ and $n_{s.a.} = 16$ for the Efficient SU(2) ansatz with 2 qubits, so at the $n = 10$ layer this approach uses 60 parameters. The number of parameters therefore increases only quadratically in the number of qubits, and polylogarithmically in the refinement depth provided by the number of statevector amplitudes. Further, assume unitaries in the Pauli decomposition (Equation (1)) of the problem Hamiltonian are implementable in polynomial depth. Then each VQE circuit has depth $O(n^2) + O(\text{poly}(n)) = O(\text{poly}(n))$ in the number of qubits, and is $O(\text{polylog}(n))$ in the refinement depth.

To seed our multigrids, and as a benchmark for our methods, we use a family of hardware-efficient ansätze called Efficient SU(2) ansätze. These are implemented in Qiskit as the class `qiskit.circuit.library.EfficientSU2`, and are primarily defined by a number of repetitions of single qubit gates—that is, $SU(2)$ unitaries—followed by CX entanglements as shown in Figure 3. This family has many of the hardware-efficient benefits discussed in Section II-B3 such as only nearest-neighbor interactions and low depth suggesting the potential for quantum advantage. We construct Efficient SU(2) circuits using the default arguments from Qiskit including the reverse linear entanglement pattern and the $\{RY, RZ\}$ gate set, but future work could explore how different choices for seed ansatz arguments affect multigrid performance on particular problems. For example, more highly expressible



(a)



(b)

Fig. 6: The Multigrid outperforms the static Efficient SU(2) ansatz for Dirichlet Laplacian VQE. In 6a, we plot the estimated eigenvalues (not the accuracy in eigenvalues) and observe that the Multigrid continues improving while the Efficient SU(2) ansatz flatlines quickly. As expected, increasing the number of shots improves performance. In 6b, we observe that Multigrid VQE uses more optimizer calls rather than giving up quickly, and has less variance in the number of optimizer calls.

seed ansatzes should have better performance; and other factors such as size or entanglement patterns could also affect performance. Our graphs are generated with 95th-percentile confidence intervals calculated by z-score. This is not meant to imply that our data is normally distributed, but to represent the uncertainty in the *mean* of the samples. Simulations were done in Python 3, where quantum circuits were implemented in Qiskit [28] and we used Scipy's optimization libraries. Given data from these simulations, Numpy and PyPlot were used for analysis and plotting. In the next two sections, we compare performance between Efficient SU(2)-seeded Multigrid VQE, Efficient SU(2) VQE, and some classical methods: Numpy for eigenvalue problems and greedy, brute-force methods for combinatorial optimization.

IV. Laplacian Eigensolver

We use multigrid VQE to approximately prepare the ground state of (and estimate the corresponding ground energy for) the discrete one-dimensional Laplacian with constant zero Dirichlet boundary conditions. By the second difference methods expanded on in Appendix A, this Hamiltonian is the $m \times m$ Toeplitz tridiagonal matrix given by

$$\nabla_D^2 = \begin{pmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{pmatrix}$$

which Sato et al. [2] give for the case where m is any power of two in a more convenient form as

$$\nabla_D^2 = S + P^\dagger S P + P^\dagger (I_0^{\otimes n-1} \otimes X) P. \tag{5}$$

Here P is the cyclic shift unitary described in more detail below, $S = I^{\otimes n-1} \otimes (I - X)$ and $I_0 = |0\rangle\langle 0| = (I + Z)/2$. Expanding these definitions allows us to express the Dirichlet Hamiltonian purely in terms of unitary-transformed Pauli strings, allowing us to implement VQE for this problem. Specifically, we have unshifted (or periodic) terms $S = 1 - X_0$ and cyclically shifted (by P) terms $S = 1 - X_0$ and

$$2^{-n+1}(1+Z_0+Z_1Z_0+Z_2Z_0+\cdots+Z_{n-2}Z_{n-1}\cdots Z_1Z_0)X_{n-1},$$

which can be optimized [3] for efficiency in any practical implementation. However, for our purposes we only need the theoretical performance of the algorithm. To speed up simulations, we use the fact that these Pauli strings all commute to calculate the whole expectation value in just one measurement.

For an increment circuit implementing the cyclic shift operation P, see Figure 5. As referenced in the caption there, this circuit can be implemented with a linear overhead in number of gates. The important part for us is that it acts on computational basis vectors by sending big-endian computational basis states $|b\rangle$ to $|\texttt{rem}(b+1, 2^n)\rangle$. For the simpler case with periodic boundary conditions, we follow Sato et al. in measuring the simpler Hamiltonian $\nabla_P^2 = S + P^\dagger S P$. (For an explicit matrix representation, see Appendix A.) This is convenient for representing the Dirichlet Hamiltonian as $\nabla_D^2 = \nabla_P^2 + P^\dagger (I_0^{\otimes n-1} \otimes X) P$, but preparing the eigenstate of ∇_P (and thus calculating its ground state energy) is not an interesting application of VQE since the state you try to get is simply the superposition of computational basis states.

This state can be prepared directly by a circuit applying a Hadamard gate to each qubit in the quantum register.

Ground state simulation for the discrete Dirichlet Laplacian, on the other hand, is more interesting. The states are less trivial than for the discrete Periodic Laplacian, but they still provide some structure to work with. In fact, the problem is well-suited to multigrid methods. This is because the ground states of these Hamiltonians seem to converge smoothly. Seeing this requires viewing them as functions from the computational basis states to $\mathbb{R}_+$. The corresponding graphs visibly form a series of increasingly smooth discrete functions with a more or less parabolic shape. This interesting structure motivates our Multigrid VQE methods and provides a good test case. For an illustration of this refinement structure and further commentary, see Figure 4. Future work could try to produce a circuit implementing a more linear interpolation better suited to this problem than our constant interpolation scheme.

The circuits in this method are polylogarithmic in the size of the obtained eigenvector. This follows from Equation (4) together with how increment circuits implementing the shift operation can be constructed with polynomially many CNOTs. This sort of benefit is common for VQE, and partially explains its popularity. We implement Multigrid VQE for Laplacian ground state preparation in Qiskit [28], and compare its estimated performance both with static application of the Efficient SU(2) ansatz and with a brute force method. We seed the Multigrid Hierarchy with an Efficient SU(2) ansatz as a proof of concept. This choice is somewhat arbitrary and does not come with a multigrid-related theoretical justification. Future work could test other seed ansätze, and seed sizes. In this case, we use an ansatz with two qubits since the discrete Dirichlet Laplacian is not defined on fewer than two qubits. There is a one qubit analog [3] given by the formula $A = 2I - X$, but this has an uninteresting ground state, the plus state. Therefore A is more analogous to ∇_P^2 than to our target of ∇_D^2. However, the ground state energy of ∇_D^2 vanishes quickly in the number of qubits and thus needs increasingly more shots to maintain its accuracy. However, this only reflects the exponential increase of the size of the multigrid resolution, and the number of shots grows more reasonably in terms of that.

A. Simulation results

To create the plots in Figure 6, we generated the multigrid hierarchy of Dirichlet Laplacians from the two to the twelve qubit case. Running each Hamiltonian through Multigrid VQE and Efficient SU(2) VQE, we estimated eigenvalue and number of optimizer calls. The solid green line represents the 10^3-shot Multigrid VQE, and the dashed green line is for the 10^3-shot Efficient SU(2) VQE. The solid blue line represents the 10^6-shot Multigrid VQE, and the dashed blue line is for the 10^6-shot Efficient SU(2) VQE. We estimate the performance of our method on Dirichlet Hamiltonians with as many as 12 qubits, corresponding to preparing eigenstates with 4,096 amplitudes. These experiments are performed using Qiskit's Aer simulator, assuming no hardware noise.

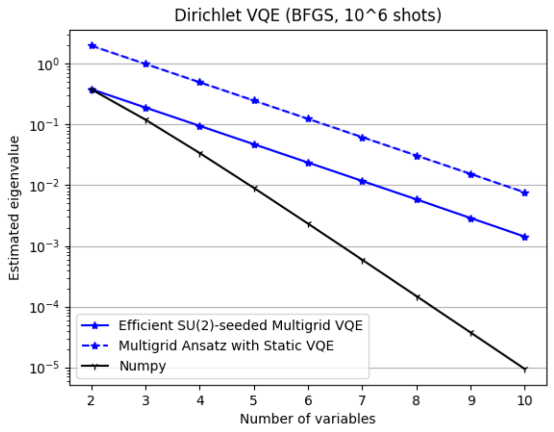
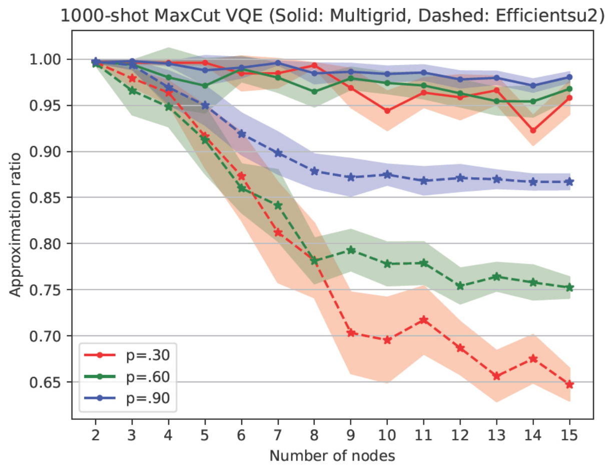


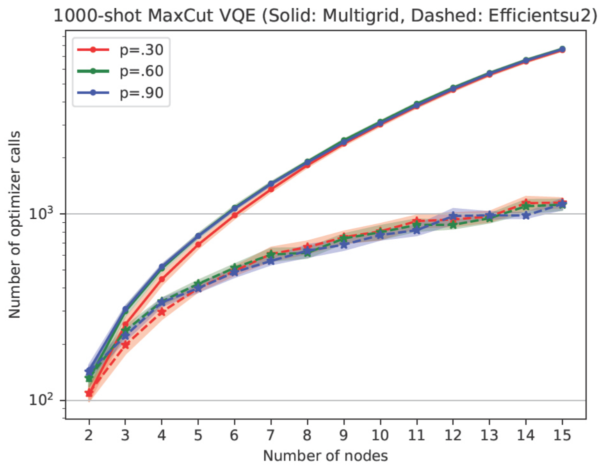
Fig. 7: In the dashed line, we test BFGS with all zero angles at every stage. In the solid line, COBYLA is used to seed a BFGS Multigrid VQE. The apparent scaling advantage suggests that gradient methods can be used to preserve good solutions once COBYLA becomes unwieldy due to number of parameters.

We observe significant improvement over the static Efficient SU(2) ansatz when using the dynamic Multigrid VQE. This result suggests that the refinement structure of the problem is being taken advantage of. In Figure 6a, the static ansätze quickly flatline in performance, while the Multigrid VQE is able to squeeze out more accuracy. Furthermore, increasing the number of shots for the Multigrid VQE roughly lines up with the heuristic prescribing $\mathcal{O}(1/\varepsilon^2)$ shots for error ε. With 10^3 shots it achieves error 10^{-2}, and with 10^6 shots, it eventually achieves error 10^{-3}. On the other hand, the static VQE does not even reach 10^{-1} accuracy with 10^3 shots, or 10^{-2} accuracy with 10^6 shots. Interestingly, 10^6 shot Multigrid VQE appears strikingly precise, producing very similar values in each test. Also, there is a crossover point after which 10^3 shot Multigrid VQE appears to outperform 10^6 shot Efficient SU(2) VQE despite its having substantially more parameters. In Figure 6b, we see that this method uses more function evaluations (optimizer calls) as a trade-off. On the other hand, it shows how Multigrid VQE is able to take more time thoroughly exploring the parameter space rather than giving up too quickly.

Figure 7 demonstrates clearly how the Multigrid VQE method of saving angles from previous rounds can lead to better solutions. The dashed line represents static application of the Multigrid Ansatz (i.e., using the ansätze produced by Algorithm 4 but with all angles initialized to zero rather than just the new ones). In contrast, the solid line represents performing the Efficient SU(2)-seeded Multigrid VQE as defined in Algorithm 3. To draw inferences about this method's ability to keep producing better results when methods such as COBYLA (Constrained Optimization by Linear Approximation) become unwieldy for large systems beyond the reach of our simulations, we use BFGS and observe that the Multigrid VQE values are consistently better than their



(a)



(b)

Fig. 8: (a) The Multigrid VQE achieves better and more stable approximation values for MaxCut than the static Efficient SU(2) ansatz does. Also, p seems to affect Efficient SU(2) performance much more drastically. (b) Multigrid VQE is able to use more optimizer calls than the Efficient SU(2) ansatz VQE, and experiences less variance in number of optimizer calls.

static counterparts.

V. Combinatorial Optimization

An instance of a combinatorial optimization problem is a set $S \subseteq \{0,1\}^n$ of bitstrings together with a function $f : S \to \mathbb{R}$ called the cost function. When $S = \{0,1\}^n$ we call the problem unconstrained, and constrained otherwise. A solution to such an instance is a bitstring $z \in S$ with $f(z) = \min_{z \in S} f(z)$. Combinatorial optimization problems are important in computer science, where they provide a framework for studying many classes of **NP** problems. In order to apply our multi-grid hierarchy ansatz to combinatorial optimization, we successively grow the problem instance by defining solution bitstring sets $S_j \subseteq \mathbf{2}^j$ and corresponding cost functions $f_j : S_j \to \mathbb{R}$ for $0 < j \leq n$ with $S = S_n$ and $f = f_n$. The exact relationship of f_i and f_{i+1} depends on the nature of the problem; for graph problems, for example, we insert the $i+1$-st vertex and its corresponding edges that connect to the already inserted first i vertices. We give two concrete examples below.

When a combinatorial optimization problem is defined using a diagonal encoding H_C of the cost function as in QAOA, this Hamiltonian representation can be used to solve the problem with VQE because the target is either the minimum eigenvalue $\lambda_0(H_C)$ or the maximum, $-\lambda_0(-H_C)$. For unconstrained problems, this is all we need; and for constrained problems we can use penalty terms to enforce the constraints.

To demonstrate these methods and test the performance of Multigrid VQE in such a setting, we implement VQE for two unconstrained optimization problems: Maximum Cut and Maximum Exact k-Satisfiability. These problems are both **NP**-hard, and future work could look into constrained versions of them such as Maximum Bisection.

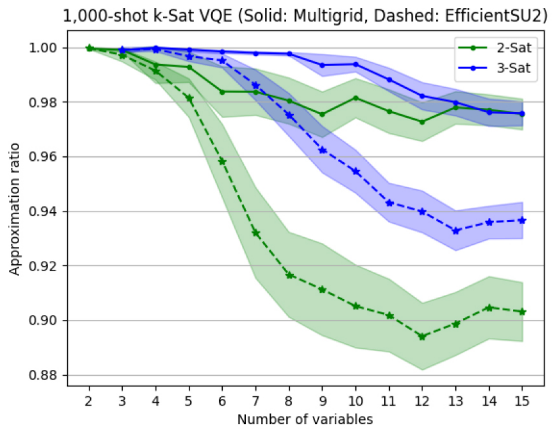
1) Maximum Cut: Given a graph $G = (V, E)$, a cut on G is a partition of V into complementary subsets $U_1 \sqcup U_2 = V$. The size of a cut is the number of edges between U_1 and U_2. The Maximum Cut (MaxCut) problem is to find the maximum size among all cuts on G. We implement Multigrid VQE for MaxCut by minimizing the inverse MaxCut Hamiltonian representation given by the weighted sum of Pauli strings

$$H_C = \frac{1}{2} \sum_{(v,w) \in E} \big(Z_v Z_w - 1\big), \tag{6}$$

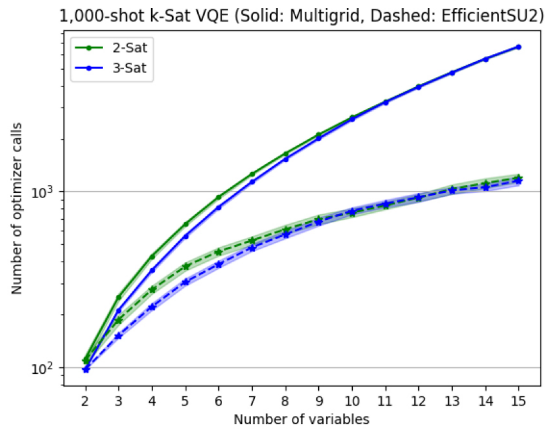
which requires only one measurement since all measurements are in the Z basis. These measurements commute so that it is enough to measure all qubits simultaneously and classically compute the ZZ terms. Letting $n = |V|$, the complexity of computing the cost function from a measurement is then $\mathcal{O}(n^2)$, since $|E| \leq |E(K_V)| = (n^2 - n)/2$. In fact, you need only measure qubits corresponding to nonzero degree nodes; but this does not affect scaling in general.

In order to use the Multigrid Hierarchy in this setting, we need a corresponding Subgraph Hierarchy to perform MaxCut on at each stage as we build the graph. We choose to naively add nodes in the order they are labeled in the NetworkX library by which they are generated. Generating Erdős-Rényi random graphs, we compare the performance of Multigrid VQE to static application of the Efficient SU(2) ansatz. To enable this, we compute the approximation ratio $C(z_{\text{obs}})/C(z_{\text{opt}})$, using a brute force method to compute $C(z_{\text{opt}})$. Here z_{obs} is the observed bitstring that we are computing the approximation ratio for and z_{opt} is an optimal bitstring.

2) Maximum k-Satisfiability: Given a propositional formula on n variables, in conjunctive normal form with exactly k variables in each of m clauses, the Maximum Exact k-Satisfiability problem (MaxEk-Sat) asks how many clauses



(a)



(b)

Fig. 9: (a) The Multigrid VQE achieves better approximation values for hard instances of Max 2 and 3-Sat than the Efficient SU(2) ansatz. Furthermore, this performance seems to be consistent across the different values of k. (b) The Multigrid VQE uses more total optimizer calls than the Efficient SU(2) ansatz, showing its better ability to explore the feasible state space.

can be satisfied simultaneously. We encode such formulae into Hamiltonians using a clause-by-clause decomposition. Given a clause, we produce the k-local Hamiltonian $I^{\otimes k} - H$ where H applies $|0\rangle\langle 0|$ to the qubits corresponding to true propositions, and $|1\rangle\langle 1|$ to the qubits corresponding to false ones. This process, derived from the De Morgan duality, penalizes bitstrings satisfying the negation of the clause.

For example, the clause $x_2 \vee \neg x_3$ maps to the 2-local Hamiltonian $I^{\otimes 3} - I \otimes |0\rangle\langle 0| \otimes |1\rangle\langle 1|$. The whole problem Hamiltonian is given by the sum of these Hamiltonians, for each clause in the formula. To convert this Hamiltonian to the Pauli-Z basis, we use the mappings $|0\rangle\langle 0| \to (I+Z)/2$ and $|1\rangle\langle 1| \to (I-Z)/2$. Expanding out the resulting tensor product yields only $\mathcal{O}(m2^k)$ terms, so it is efficient in terms of m and n. This means that our example becomes

$$I^{\otimes 3} - \frac{I^{\otimes 3} - I^{\otimes 2} \otimes Z + I \otimes Z \otimes I - I \otimes Z^{\otimes 2}}{4}.$$

All Pauli-Z measurements commute, so we can optimize the circuit to only use one circuit execution per shot. Furthermore, we only need to measure qubits corresponding to variables that occur in some term (although this does not matter practically).

We produce a Multigrid Hierarchy for this problem by adding variables in the order they are generated by our code. When a variable is added, all clauses including that variable or variables already added are admitted. However, we do not admit any clause involving any variable not yet added.

We compare Multigrid Random MaxEk-Sat VQE to static Efficient SU(2) VQE, using a brute force algorithm to compute exact solutions for determining approximation ratios. Following Golden et al. [29], we generate hard instances beyond the difficulty phase transition by setting $m = 3n$ for the $k = 2$ case and $m = 6n$ for the $k = 3$ case.

A. Simulation Results

As in the previous section, we run our tests with Qiskit's Aer simulator, assuming that there is no hardware noise.

1) Maximum Cut: To create the plots in Figure 8, we prepared Erdős-Rényi random graphs on 15 nodes and generated their multigrid hierarchies. Running each subgraph through Multigrid VQE and Efficient SU(2) VQE, we can see how the methods diverge in performance before reaching their solutions. Some subgraphs had MaxCut zero (due to having no edges), and those datapoints were thrown away for Figure 8a since their approximation ratios are undefined.

For MaxCut, we expect saving information about subgraph MaxCut to inform the optimization in subsequent refinement stages and improve performance. Using NetworkX, we generated 15-node Erdős-Rényi random graphs with edge probabilities $p \in \{0.3, 0.6, 0.9\}$ and compare approximation ratios between the Efficient SU(2)-seeded Multigrid VQE and the Efficient SU(2) VQE.

2) Maximum k-Satisfiability: To create the plots in Figure 9, we prepared random Ek-SAT instances at the satisfiability threshold for 15 variables and generated their multigrid hierarchies. Running each sub-formula through Multigrid VQE and Efficient SU(2) VQE, we can see how the methods diverge in performance before reaching their solutions. It is worth noting that some sub-formulas had MaxSat zero (due to every clause sharing some not-yet-added variable) and that those datapoints were thrown away since their approximation ratios are undefined. Further, there is no data for 2 variables since there is no E3-SAT instance with only 2 variables.

For Max Ek-SAT, we expect the increased parameterization of the Multigrid Hierarchy and saving information about sub-formula MaxSat to inform the optimization in subsequent refinement stages and improve performance. This appears to be achieved, as in Figure 9 the multigrid methods outperform their hardware-efficient counterparts.

VI. Conclusion

We have presented the hierarchical multi-grid ansatz as an alternative approach for the variational quantum eigensolver. Our approach generalizes classical mesh refinement, and can be applied anywhere a hierarchy of problems can be defined. Using a particular seed ansatz and multigrid refinement method, we showed improved average performance by comparison with the Efficient SU(2) VQE for the Laplacian eigenvalue problem, as well as for MaxCut and Max-Ek-Sat.

Future work: We will study this ansatz further for other application domains such as quantum chemistry, where physical Hamiltonians such as the molecular electronic Hamiltonian could perhaps gain from exploiting the structure or symmetries of molecules in a multigrid hierarchy. Additional studies remain to be done to compare the multigrid ansatz for optimization problems to other quantum optimization algorithms, such as QAOA variants and filtered VQE. Also, in the future, we plan to complement our numerical studies with actual hardware studies on existing NISQ devices, and to explore more efficient ansatz designs such as ansaetze tailored to qubit architectures without all-to-all connectivity.

Appendix

In this appendix, we present the derivation of the Discrete Laplacian Hamiltonians. To begin with, Poisson's equation $\nabla^2 f = g$ can be discretized by second difference methods. Assuming the one-dimensional case with unit displacement, we have the approximation

$$g_i = \nabla^2 f_i = \partial f/\partial x \approx -f_{i-1} + 2f_i - f_{i+1}.$$

In terms of vector entries, this translates to the formula

$$\sum_{i=0}^{n}(-f_{i-1} + 2f_i - f_{i+1})\mathbf{e}_i \approx \nabla^2 f \approx \sum_{i=0}^{n} g_i \mathbf{e}_i,$$

where, taking Dirichlet boundary $f_{-1} = f_{n+1} = 0$, the left-hand side motivates the discrete Dirichlet Laplacian. This is the linear operator

$$\nabla_D^2 = \begin{pmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{pmatrix}$$

producing this matrix from the vector f. Using periodic boundary $f_{-1} = f_{n-1}$, $f_n = f_0$, we instead get the discrete periodic Laplacian

$$\nabla_P^2 = \begin{pmatrix} 2 & -1 & & & & & -1 \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ -1 & & & & & -1 & 2 \end{pmatrix}$$

which can be seen by matrix multiplication to satisfy the relationship $\nabla_D^2 = \nabla_P^2 + P^\dagger(I^{\otimes n-1} \otimes X)P$ from Sato et al. [2]. The same paper gives a third Laplacian for Neumann boundary conditions. Assuming zero derivative at the boundary, the second difference approximation changes to

$$g_0 = \nabla^2 f_0 = \partial f/\partial x \approx f_{-1} - f_1$$
$$g_n = \nabla^2 f_n = \partial f/\partial x \approx f_n - f_{n-1}$$

so that the discrete Laplacian operator is given by

$$\nabla_N^2 = \begin{pmatrix} 1 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 1 \end{pmatrix}$$

completing the set of Laplacians from Sato et al. [2]. Future work could look at adapting these methods to more complicated boundary conditions.

References

[1] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, no. 1, p. 4213, Jul. 2014.

[2] Y. Sato, R. Kondo, S. Koide, H. Takamatsu, and N. Imoto, "Variational quantum algorithm based on the minimum potential energy for solving the Poisson equation," *Physical Review A*, vol. 104, no. 5, p. 052409, Nov. 2021.

[3] H.-L. Liu, Y.-S. Wu, L.-C. Wan, S.-J. Pan, S.-J. Qin, F. Gao, and Q.-Y. Wen, "Variational quantum algorithm for the Poisson equation," *Physical Review A*, vol. 104, p. 022418, Aug. 2021.

[4] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, and J. Tennyson, "The Variational Quantum Eigensolver: A review of methods and best practices," *Physics Reports*, vol. 986, pp. 1–128, Nov. 2022.

[5] A. Jayakumar, A. Adedoyin, J. Ambrosiano, P. Anisimov, W. Casper, G. Chennupati, C. Coffrin, H. Djidjev, D. Gunter, S. Karra, N. Lemons, S. Lin, A. Malyzhenkov, D. Mascarenas, S. Mniszewski, B. Nadiga, D. O'Malley, D. Oyen, S. Pakin, L. Prasad, R. Roberts, P. Romero, N. Santhi, N. Sinitsyn, P. J. Swart, J. G. Wendelberger, B. Yoon, R. Zamora, W. Zhu, S. Eidenbenz, A. Bärtschi, P. J. Coles, M. Vuffray, and A. Y. Lokhov, "Quantum Algorithm Implementations for Beginners," *ACM Transactions on Quantum Computing*, vol. 3, no. 4, pp. 18:1–18:92, Jul. 2022.

[6] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, "An adaptive variational algorithm for exact molecular simulations on a quantum computer," *Nature Communications*, vol. 10, no. 1, p. 3007, Jul. 2019.

[7] X. Liu, A. Angone, R. Shaydulin, I. Safro, Y. Alexeev, and L. Cincio, "Layer VQE: A Variational Approach for Combinatorial Optimization on Noisy Quantum Computers," *IEEE Transactions on Quantum Engineering*, vol. 3, no. 01, pp. 1–20, Jan. 2022.

[8] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, "Variational Quantum Linear Solver," *Quantum*, vol. 7, p. 1188, Nov. 2023. [Online]. Available: https://doi.org/10.22331/q-2023-11-22-1188

[9] M. Illa and M. J. Savage, "Basic elements for simulations of standard-model physics with quantum annealers: Multigrid and clock states," *Physical Review A*, vol. 106, no. 5, p. 052605, Nov. 2022.

[10] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, "Variational quantum algorithms for nonlinear problems," *Phys. Rev. A*, vol. 101, p. 010301, Jan 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.101.010301

[11] A. Abbas, A. Ambainis, B. Augustino, A. Baertschi, H. Buhrman, C. J. Coffrin, G. Cortiana, V. Dunjko, D. J. Egger, B. G. Elmegreen, N. Franco, F. Fratini, B. Fuller, J. Gacon, C. Gonciulea, S. Gribling, S. Gupta, S. Hadfield, R. Heese, G. Kircher, T. Kleinert, T. Koch, G. Korpas, S. Lenk, J. Marecek, V. Markov, G. Mazzola, S. Mensa, N. Mohseni, G. Nannicini, C. O'Meara, E. Peña Tapia, S. Pokutta, M. Proissl, P. Rebentrost, E. Sahin, B. C. B. Symons, S. Tornow, V. Valls, S. Woerner, M. L. Wolf-Bauwens, J. Yard, S. Yarkoni, D. Zechiel, S. Zhuk, and C. Zoufal, "Quantum optimization: Potential, challenges, and the path forward," 12 2023. [Online]. Available: https://www.osti.gov/biblio/2229681

[12] B. C. B. Symons, D. Galvin, E. Sahin, V. Alexandrov, and S. Mensa, "A practitioner's guide to quantum algorithms for optimisation problems," *Journal of Physics A: Mathematical and Theoretical*, vol. 56, no. 45, p. 453001, oct 2023. [Online]. Available: https://dx.doi.org/10.1088/1751-8121/ad00f0

[13] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, Aug. 2021.

[14] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," Nov. 2014.

[15] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, "Filtering variational quantum algorithms for combinatorial optimization," *Quantum Science and Technology*, vol. 7, no. 1, p. 015021, feb 2022. [Online]. Available: https://dx.doi.org/10.1088/2058-9565/ac3e54

[16] D. J. Egger, J. Mareček, and S. Woerner, "Warm-starting quantum optimization," *Quantum*, vol. 5, p. 479, Jun. 2021. [Online]. Available: https://doi.org/10.22331/q-2021-06-17-479

[17] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, "Obstacles to variational quantum optimization from symmetry protection," *Phys. Rev. Lett.*, vol. 125, p. 260505, Dec 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.125.260505

[18] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, "Cost function dependent barren plateaus in shallow parametrized quantum circuits," *Nature Communications*, vol. 12, no. 1, p. 1791, Mar. 2021.

[19] S. Boulebnane and A. Montanaro, "Solving boolean satisfiability problems with the quantum approximate optimization algorithm," Aug. 2022.

[20] J. Golden, A. Bärtschi, S. Eidenbenz, and D. O'Malley, "Numerical Evidence for Exponential Speed-up of QAOA over Unstructured Search for Approximate Constrained Optimization," in *IEEE International Conference on Quantum Computing and Engineering QCE'23*, Sep. 2023.

[21] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem," Jun. 2015.

[22] B. Barak, A. Moitra, R. O'Donnell, P. Raghavendra, O. Regev, D. Steurer, L. Trevisan, A. Vijayaraghavan, D. Witmer, and J. Wright, "Beating the Random Assignment on Constraint Satisfaction Problems of Bounded Degree," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques APPROX/RANDOM'15*, Aug. 2015, pp. 110–123.

[23] W. Ritz, "Über eine neue Methode zur Lösung gewisser Variationsprobleme der mathematischen Physik," *Journal für die reine und angewandte Mathematik*, 1909.

[24] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum computation by adiabatic evolution," Jan. 2000.

[25] 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. 549, no. 7671, pp. 242–246, Sep. 2017.

[26] L. Leone, S. F. Oliviero, L. Cincio, and M. Cerezo, "On the practical usefulness of the Hardware Efficient Ansatz," Nov. 2022.

[27] C. Gidney, "Constructing large controlled nots / Constructing large increment gates / Using quantum gates instead of ancilla bits," Jun 2015. [Online]. Available: https://algassert.com/circuits/2015/06/22/Using-Quantum-Gates-instead-of-Ancilla-Bits.html

[28] Qiskit contributors, "Qiskit: An open-source framework for quantum computing," 2023.

[29] J. Golden, A. Bärtschi, D. O'Malley, and S. Eidenbenz, "The Quantum Alternating Operator Ansatz for Satisfiability Problems," in *IEEE International Conference on Quantum Computing and Engineering QCE'23*, Sep. 2023.