Supplementary Information

A Quantum Approximate Optimization Method For Finding Hadamard Matrices

ANDRIYAN BAYU SUKSMONO, Institut Teknologi Bandung, Indonesia

1 REPRESENTATIVE PROBLEMS FOR EACH k-BODY CASES

The general form of Hamiltonian for the Hadamard Matrix Search problem is given by Eq.(7) in the main paper as follows:

$$\hat{H}_C = \sum_j c_j \hat{\sigma}_j^z + \sum_{j,k} c_{jk} \hat{\sigma}_j^z \hat{\sigma}_k^z + \sum_{j,k,m} c_{jkm} \hat{\sigma}_j^z \hat{\sigma}_k^z \hat{\sigma}_m^z + \sum_{j,k,m,n} c_{jkmn} \hat{\sigma}_j^z \hat{\sigma}_k^z \hat{\sigma}_m^z \hat{\sigma}_n^z$$

In this equation, the first term represents the one-body interaction, the second term corresponds to the two-body interaction, and the third and fourth terms represent higher-order interactions, specifically the three-body and four-body interactions. In the case of D-Wave, the one-body and two-body interactions can be implemented directly, while the higher-order terms require additional ancillary qubits. In contrast, gate-based quantum computers can implement both low-order and high-order interaction terms directly. In the followings, we explain how each Hamiltonian term is mapped into quantum circuits of gate-based quantum computers.

To validate our implementation, we evaluated each k-body circuit by first solving a simplified problem: finding low-order Hadamard matrices. We specifically tested our circuits on the prototypical problem of finding second-order Hadamard matrices, gradually increasing the number of unknown elements to simulate the growing complexity of interaction terms as the k-body order increases. These tests were performed using a quantum computer simulator before applying the circuits to the full problem.

1.1 Case-1: 1-body interaction problem

The first case to consider involves only 1-body terms. A representative problem of finding a second-order H-matrix with one unknown element can be expressed as follows:

$$\begin{pmatrix} 1 & 1 \\ 1 & s_1 \end{pmatrix}$$

The energy function of this problem is then given by

$$E = 2(1 + s_1) \tag{S.1}$$

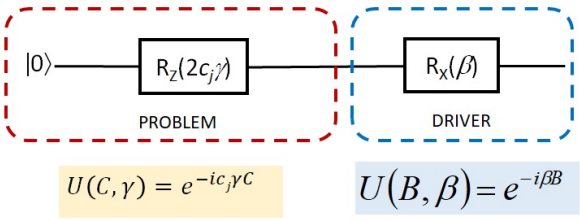
whose corresponding Hamiltonian is

$$\hat{H}(\hat{\sigma}) = 2\left(1 + \hat{\sigma}_1^z\right) \tag{S.2}$$

It is obvious that the solution will be $s_1 = -1$, which corresponds to "1" in the binary variable domain (recall that the spin-to-binary variable transformation is: ”0” $\leftrightarrow$ 1 and ”1” $\leftrightarrow -1$). The corresponding QAOA circuit can be constructed after insertion of driver Hamiltonian, which represented as an X-rotation of angle β displayed in Fig.S.1(a). After constructing the circuit and running the program, we obtain the results shown in Fig. S.1 (b) and (c).

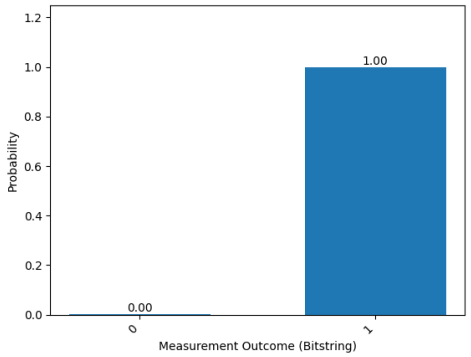
We can calculate the performance of this algorithm as follows. Since there is only one variable, a random algorithm will output a single bit string, therefore, there are two possibilities of the output, which are "0" and "1". The probability of the solution in the random algorithm is $P_R = \frac{1}{2} = 0.5$. On

Author’s address: Andriyan Bayu Suksmono, suksmono@itb.ac.id, Institut Teknologi Bandung, Jl. Ganesha No.10, Bandung, Indonesia.



(a) A quantum circuit for 2-qubits QAOA problem

(b) Realized Quantum Circuit in Qiskit



(c) Distribution of solution

Fig. S.1. Quantum Circuit for 1-body Interaction: (a) design of the quantum circuit, (b) implemented quantum circuit in qiskit, (c) distribution of the solutions

the other hand, the quantum algorithm gives one correct answer so that $P_A = 1.0$ as shown in the histogram. Accordingly, we have $xRAR = \frac{P_A}{P_R} = 2$, which is the maximum value of this problem.

1.2 Case-2: 2-body interaction problem

The prototypical problem for a 2-body case can be formulated similarly, but now we have two variables s_1, s_2. We can formulate the searched matrix as follows

$$\begin{pmatrix} 1 & s_1 \\ 1 & s_2 \end{pmatrix}$$

The energy function related to this problem is

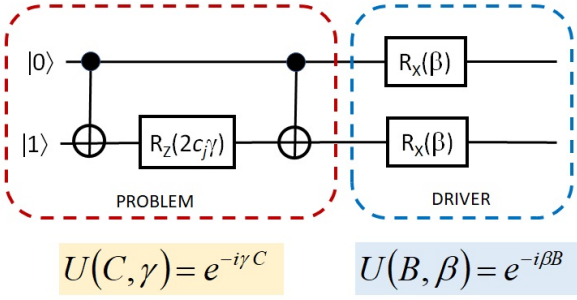
$$E = (s_1 + s_2)^2 = 2(1 + s_1 s_2) \tag{S.3}$$

whose corresponding Hamiltonian is

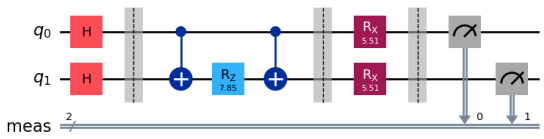
$$\hat{H}(\hat{\sigma}) = 2\left(1 + \hat{\sigma}_1^z \hat{\sigma}_2^z\right) \tag{S.4}$$

the simplification is obtained since we know that $s_i \in \{-1, 1\}$ implying $s_i^2 = 1$. Obviously, the solution in the binary form will be either "01" or "10". The quantum circuit for this problem is given by S.2 (a).

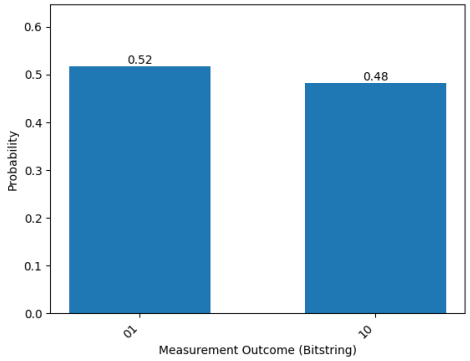
Running the circuit on a quantum processor simulator produces the results shown in Fig.S.2 (b), with the corresponding histogram displayed in (c). The histogram shows that the solutions are almost evenly distributed between "01" and "10", as anticipated. Since the output is a 2-bit



(a) A quantum circuit for 2-qubits QAOA problem



(b) Realized Quantum Circuit in Qiskit



(c) Distribution of solution

Fig. S.2. QAOA circuits and simulation results of 2-qubits problem: (a) design of the quantum circuit, (b) implemented quantum circuit in qiskit, (3) distribution of the solutions

strings and there are 2 valid solutions, the probability of the solution by the random algorithm is $P_R = \frac{2}{4} = \frac{1}{2} = 0.5$. From Fig.S.2 (c) we have $P_A = 0.52 + 0.48 = 1.0$; therefore $xRAR = \frac{1}{0.5} = 2$.

1.3 Case-3: 3-body interaction problem

The 3-body prototypical problem can be represented by 3-unknown entries with corresponding variables s_1, s_2, s_3 in the 2-order H-matrix as follows

$$\begin{pmatrix} 1 & s_1 \\ s_2 & s_3 \end{pmatrix}$$

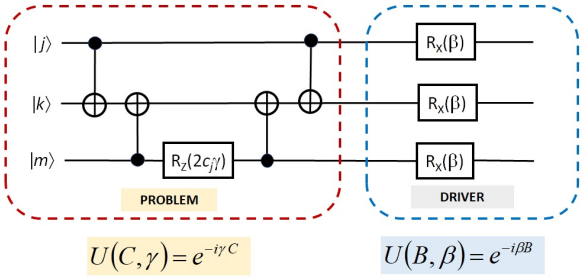
The related energy function can be computed similarly, which gives the following expression

$$E = (s_1 + s_2 s_3)^2 = 2(1 + s_1 s_2 s_3) \tag{S.5}$$

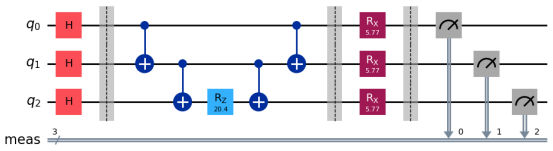
whose corresponding Hamiltonian is

$$\hat{H}(\hat{\sigma}) = 2\left(1 + \hat{\sigma}_1^z \hat{\sigma}_2^z \hat{\sigma}_3^z\right) \tag{S.6}$$

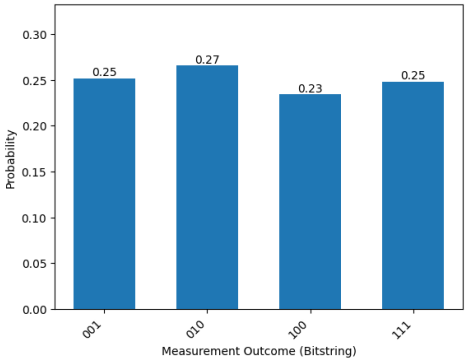
The quantum circuit for this problem is displayed in Fig. S.3 (a). Then, we run the circuit in quantum simulator and get the following results shown in Fig.S.3 (b) and (c). The histogram in (c) shows the peak distributions into 4 cases corresponding with solution of "001", "010", "100", and "111".



(a) A quantum circuit for 3-qubits QAOA problem



(b) Realized Quantum Circuit in Qiskit



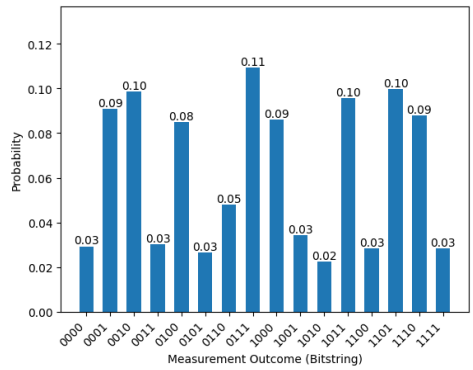
(c) Distribution of solution

Fig. S.3. QAOA circuits and simulation results of 3-qubits problem: (a) design of the quantum circuit, (b) implemented quantum circuit in qiskit, (3) distribution of the solutions

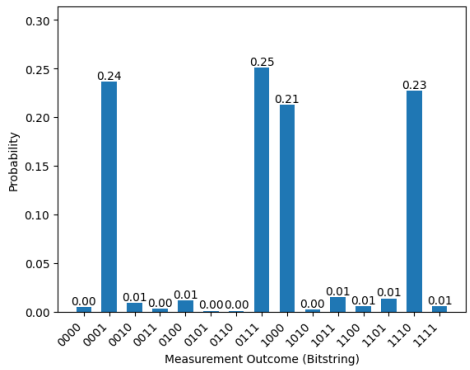
Considering bit "1" represents "-1" spin-value, all of the peaks correspond to correct solutions. Since the output is a 3-length bit strings, the probability of the solution in a random algorithm is $P_R = 4\frac{1}{2^3} = \frac{1}{2} = 0.5$, whereas the probability of the solution by the quantum algorithm as shown in the histogram is $P_A = 0.25 + 0.27 + 0.23 + 0.75 = 1.0$. Therefore, the performance metric is $xRAR = \frac{1.0}{0.5} = 2.0$.

2 CASCADED k-BODY CASES

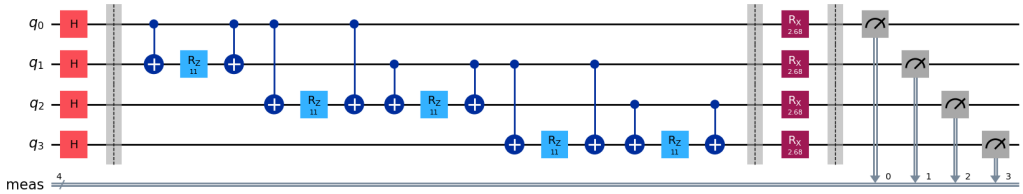
In the previous section, we presented experimental results for individual k-body cases, which are simplest scenarios involving a Problem Hamiltonian with a single term. Now, we present the results for cases where multiple terms are combined. A few sub cases will be considered, which are: (a) uniform k-body and with uniform coefficients, (b) uniform k-body with non-uniform coefficients, and (c) mixed k-body. We minimize the mean error of the objective function by using the COBYLA (Constrained Optimization BY Linear Approximation) method provided in Python package. The initial values of the parameters $\{\gamma, \beta\}$ are set at random.



(a) Distribution of solution for $p = 1$



(b) Distribution of solution for $p = 4$



(c) Quantum Circuit $p = 1$

Fig. S.4. Uniform 2-Body with Uniform Coefficients: (a) solution distribution of number of layer $p = 1$, (b) solution distribution of higher number of layer $p = 4$, (c) implemented quantum circuit using qiskit

2.1 Case-1: 2-body terms with uniform coefficients

The Hamiltonian terms in this case are similar to the max-cut problem that can be found in various quantum computing examples. Here, we define a Hamiltonian that is not related particularly to the Hadamard matrix problem. All of the terms are 2-body interactions, which is given by

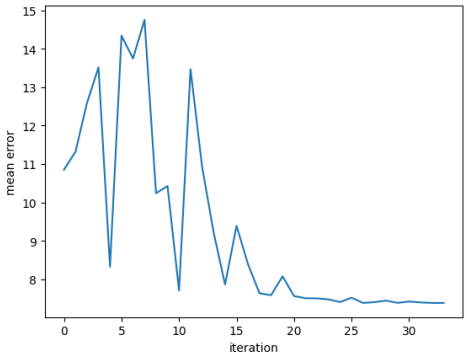
$$\hat{H}(\hat{\sigma}) = \hat{\sigma}_0^z\hat{\sigma}_1^z + \hat{\sigma}_0^z\hat{\sigma}_2^z + \hat{\sigma}_1^z\hat{\sigma}_2^z + \hat{\sigma}_1^z\hat{\sigma}_3^z + \hat{\sigma}_2^z\hat{\sigma}_3^z - 1 \tag{S.7}$$

Since there are 4 binary variables, the number of total possible output is 16. An exhaustive check shows that there are 4 bit strings of correct solutions, which are $\{0001, 0111, 1000, 1110\}$. This give the probability of random algorithm $P_R = \frac{4}{16} = \frac{1}{4}$.

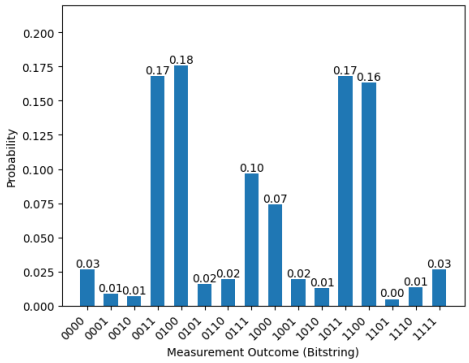
We found that running QAOA on a simulator with $p = 1$ is insufficient to find all solutions, when relying only to the dominant (highest) probability value. We got a better result when the number

of layer was increased to $p = 4$. The simulation results are displayed in Fig.S.4, with (a) distribution of solution for $p = 1$, (b) distribution of solution for $p = 4$, and (c) quantum circuit of the problem.

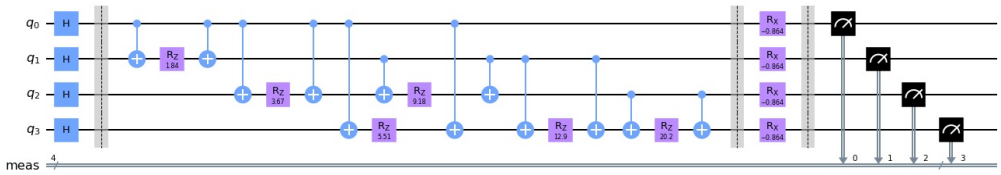
For $p = 4$, we obtain $P_A = 0.24 + 0.25 + 0.21 + 0.23 = 0.93$, therefore the performance in term of xRAR is $xRAR = \frac{P_A}{P_R} = 3.72$.



(a) Iteration curve for $p = 1$



(b) Distribution of solution for $p = 1$



(c) Quantum Circuit $p = 1$

Fig. S.5. Uniform 2-Body with Non-Uniform Coefficients: (a) Iteration curve for number of layer $p = 1$. The residual error still relatively high, (b) distribution of solution for $p = 1$. A number of peaks emerge, not all are expected solutions, (c) implemented quantum circuit on qiskit

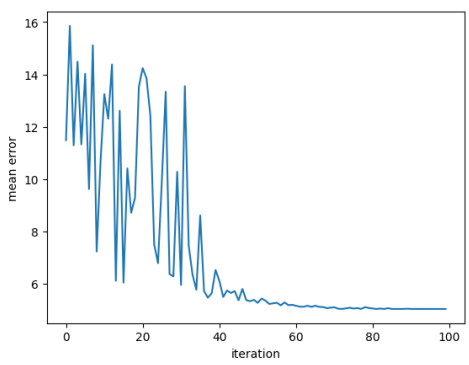
2.2 Case-2: 2-body with non-uniform coefficients

In this experiment, the Hamiltonian of the problem is given by the following

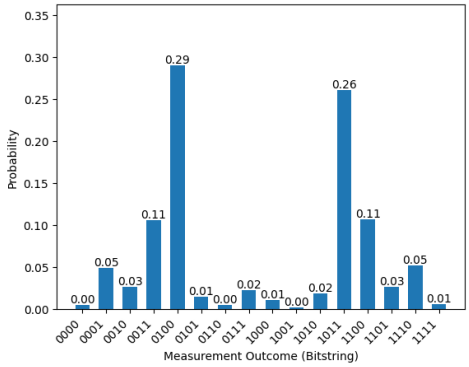
$$\hat{H}(\hat{\sigma}) = \hat{\sigma}_0^z\hat{\sigma}_1^z + 2\hat{\sigma}_0^z\hat{\sigma}_2^z + 3\hat{\sigma}_0^z\hat{\sigma}_3^z + 5\hat{\sigma}_1^z\hat{\sigma}_2^z + 7\hat{\sigma}_1^z\hat{\sigma}_3^z + 11\hat{\sigma}_2^z\hat{\sigma}_3^z - 3 \tag{S.8}$$

The constant 3 in this equation is an offset that makes the absolute minimum value of H_P equal to zero. Eq. (S.7) differs slightly from Eq. (S.8); while both involve only two-body terms, the latter includes various different coefficients. The total number of possible output is also 16, but the correct solutions is only 2; which are $\{0100, 1011\}$, and this gives $P_R = \frac{1}{8}$.

We do the experiment by increasing the number of layers p. The results are displayed in Fig. S.5 and Fig. S.6, which shows increasing performance in terms of increasing xRAR and decreasing objective error. The table in Fig. S.6 (c) clearly shows increasing performance with increasing number of layers.



(a) Iteration curve for $p = 4$



(b) Distribution of solution for $p = 4$

No	N-layer p	Obj. Error	xRAR
1	1	8.621	3.656
2	4	5.068	5.320
3	8	2.064	6.586
4	16	0.920	7.461
5	32	0.066	7.953

(c) Table of performance

Fig. S.6. Uniform 2-Body with Non-Uniform Coefficients: high layer number and table of performance: (a) iteration curve for $p = 4$, showing less residual error than that of with $p = 1$, (b) distribution of solutions with 2 dominant peaks, which correspond exactly with expected solutions, (c) performance table showing reduced error and increased xRAR performance with increasing number of layers p

3 NOISE ANALYSIS AND ERROR MITIGATION

To address the discrepancy between ideal simulations and actual quantum hardware results, we conducted additional simulations incorporating realistic noise models provided by Qiskit. These noisy simulations aim to approximate the behavior of IBM quantum processors and to quantify the impact of noise on algorithm performance. Specifically, we tested the WBH-based QAOA of order 12/36 instance using a quantum simulator with a depolarizing noise model: 0.1% error was applied to all single-qubit gates (H, X, Y, Z, Rx, Ry, Rz, U1, U2, U3), and 2% error was applied to two-qubit gates (e.g., CX, CZ, SWAP, ISWAP). The comparison between ideal and noisy outcomes is visualized in Fig.S.7.

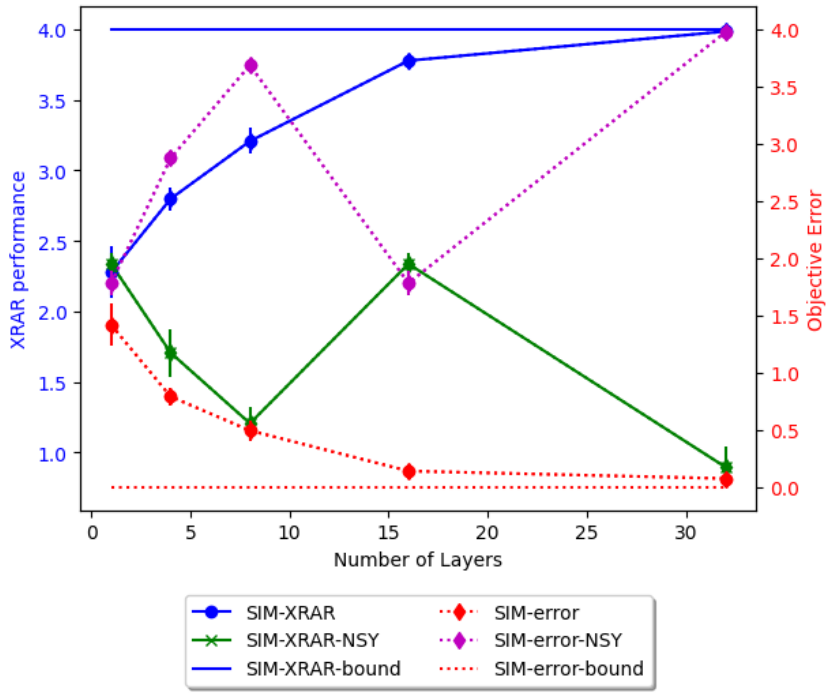


Fig. S.7. Comparison of xRAR metric and error rate between ideal simulation and noisy simulation for the WBH-12 instance. The noisy simulation includes 0.1% single-qubit and 2% two-qubit depolarizing noise.

The results show notable performance degradation at higher circuit depths due to decoherence and gate errors. Interestingly, under moderate noise levels used in our simulations, we observed a shallow-depth region (at $p = 16$) where the xRAR metric showed a small increase and the error decreased, indicating a potential local optimum in the noise-performance tradeoff. A similar behavior emerged in real-device experiments, where the xRAR metric peaked at $p = 3$, before deteriorating at greater depths, as shown in Fig. 8(b) in the main paper. These trends are expected to generalize to higher-order Hadamard matrix search instances.

Several error mitigation techniques have been proposed to counter the limitations of Noisy Intermediate-Scale Quantum (NISQ) devices. Within the IBM/Qiskit ecosystem, some advanced tools are currently only available through commercial packages. Among the notable solutions, Q-CTRL's *Fire Opal* employs error suppression during circuit execution, significantly improving fidelity and robustness at larger depths. *Operator Backpropagation* (OBP), developed in collaboration with

QunaSys, reduces effective circuit depth by analytically shifting operations into the measurement stage, thereby minimizing noise accumulation. Additionally, *Sample-based Quantum Diagonalization* (SQD) by Multiverse Computing post-processes noisy measurement data to enhance eigenvalue accuracy. These techniques help stabilize xRAR values and mitigate total error growth, although none can fully recover the ideal performance of a noiseless quantum circuit.

4 CONSTRUCTION OF THE QUANTUM CIRCUIT FROM HAMILTONIAN

The following pseudocode outlines the construction of a quantum circuit based on the given Hamiltonian terms. The algorithm takes as input the problem Hamiltonian, the number of Hamiltonian terms, a vector of angle parameters (β and γ), the number of qubits, and the number of layers. The output is the resulting quantum circuit. A Python implementation of this algorithm is available on GitHub.

Algorithm 1 Construct quantum circuit from Hamiltonian

```
# Input:
#    H: hamiltonian of the problem with term H[0], H[1], ...
#    n_terms: number of terms in the Hamiltonian
#    β, γ: vector of parameters
#    n_qubits: number of qubits,
#    qubits: qubit[0], qubit[1], ..., qubit[n_qubits − 1]
#    n_layers: number of qaoa layers
# Output:
#    qc ← quantum circuit
# prepare the quantum circuit
qc ← create quantum circuit(n_qubits)
# using hadamard-gate, set qubits to equal superposition of |0⟩ and |1⟩
for m = 0 to n_qubits − 1 do
    qc.qubit[m] ← hadamard(qc.qubit[m])
end for
for m = 0 to n_layers − 1 do        # construct n_layers qaoa quantum circuit
    for n = 0 to n_terms − 1 do        # Calculate Problem Hamiltonian
        # eg. H = x_0 + 2x_0x_1 + 3x_0x_1x_2 − 4x_0x_1x_2x_3
        coeff = coefficients of (H[n])        # coefficients of (H[3]) = −4
        order = order of (H[n])        # order of (H[3]) = 4
        # list of variables (H[3]) = [0, 1, 2, 3]
        # qlist = list of interacting qubits (H[n]) # var list
        switch order(H[n]) do
            case 1
                qc ← qc.rz(2*coeff*gamma[m], qubit[qlist[0]])
            case 2
                qc ← qc.cx(qubit[qlist[0]], qubit[qlist[1]])
                qc ← qc.rz(2*coeff*gamma[m], qubit[qlist[1]])
                qc ← qc.cx(qubit[qlist[0]], qubit[qlist[1]])
            case 3
                qc ← qc.cx(qubit[qlist[0]], qubit[qlist[1]])
                qc ← qc.cx(qubit[qlist[1]], qubit[qlist[2]])
                qc ← qc.rz(2*coeff*gamma[m], qubit[qlist[2]])
                qc ← qc.cx(qubit[qlist[1]], qubit[qlist[2]])
                qc ← qc.cx(qubit[qlist[0]], qubit[qlist[1]])
            case 4
                qc ← qc.cx(qubit[qlist[0]], qubit[qlist[1]])
                qc ← qc.cx(qubit[qlist[1]], qubit[qlist[2]])
                qc ← qc.cx(qubit[qlist[2]], qubit[qlist[3]])
                qc ← qc.rz(2*coeff*gamma[m], qubit[qlist[3]])
                qc ← qc.cx(qubit[qlist[2]], qubit[qlist[3]])
                qc ← qc.cx(qubit[qlist[1]], qubit[qlist[2]])
                qc ← qc.cx(qubit[qlist[0]], qubit[qlist[1]])
        end switch
    end for
    for k = 0 to n_qubits do    # Calculate Mixing Hamiltonian
        qc ← rotation_x(2*beta[m],qubit[k])
    end for
end for
```

5 PERFORMANCE COMPARISON OF COBYLA AND NELDER-MEAD OPTIMIZERS

To evaluate the effect of optimizer selection on the performance –with the WBH-Based QAOA method as an example, we conducted a comparative analysis of two commonly used optimization methods, i.e the COBYLA and Nelder-Mead. While COBYLA was used in the main experiments, this additional comparison aims to examine the potential benefits or limitations of using alternative optimizers in the context of the WBH-based QAOA search problem. Both optimizers were tested under identical conditions for the WBH-12/36 cases, with each experiment repeated 30 times. The results, shown in Figure S.8, demonstrate the performance differences in terms of xRAR metric and error rate, providing insights into the optimizer's influence on circuit outcomes

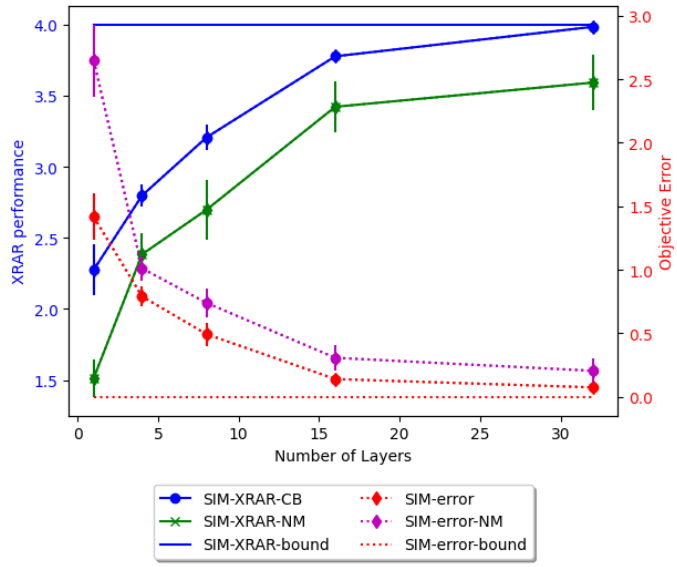


Fig. S.8. Comparison of optimization performance between COBYLA (CB) and Nelder-Mead (NM) methods on the WBH-based QAOA method for finding order 12/36 Hadamard matrices.

In Fig. S.8, the xRAR metric (left vertical axis, in blue and green) and the objective error (right vertical axis, in red and purple) are plotted against the number of circuit layers. This figure shows that COBYLA consistently achieves higher xRAR values and lower errors than Nelder-Mead across most layer depths. Error bars represent confidence interval over 30 trials.