Huffman-Code-Based Ternary Tree Transformation

Qing-Song Li[1,2], Huan-Yu Liu[1,2], Qingchun Wang[3*], Yu-Chun Wu[1,2,3*], and Guo-Ping Guo[1,2,3,4]

[1]*Laboratory of Quantum Information, University of Science and Technology of China, Hefei 230026, China*
[2]*Anhui Province Key Laboratory of Quantum Network, University of Science and Technology of China, Hefei 230026, China*
[3]*Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230026, China*
[4]*Origin Quantum Computing, Hefei 230026, China*

(Received 14 May 2025; accepted manuscript online 6 August 2025)

Using a quantum computer to simulate fermionic systems requires fermion-to-qubit transformations. Usually, lower Pauli weight of transformations means shallower quantum circuits. Therefore, most existing transformations aim for lower Pauli weight. However, in some cases, the circuit depth depends not only on the Pauli weight but also on the coefficients of the Hamiltonian terms. In order to characterize the circuit depth of these algorithms, we propose a new metric called weighted Pauli weight, which depends on Pauli weight and coefficients of Hamiltonian terms. To achieve smaller weighted Pauli weight, we introduce a novel transformation, Huffman-code-based ternary tree (HTT) transformation, which is built upon the classical Huffman code and tailored to different Hamiltonians. We tested various molecular Hamiltonians and the results show that the weighted Pauli weight of the HTT transformation is smaller than that of commonly used mappings. At the same time, the HTT transformation also maintains a relatively small Pauli weight. The mapping we designed reduces the circuit depth of certain Hamiltonian simulation algorithms, facilitating faster simulation of fermionic systems.

DOI: 10.1088/0256-307X/42/10/100001 CSTR: 32039.14.0256-307X.42.10.100001

1. Introduction. Quantum computers, due to their inherent parallelism, hold potential advantages over classical computers in certain problems. Currently, quantum computers are being utilized for tasks like solving numerical equations,[1–3] quantum chemistry,[4,5] simulating physical systems,[6–8] and other areas.[9–14] Among them, simulating fermionic Hamiltonians is an important research field. Hamiltonian simulation refers to implementing the time evolution operator $\mathrm{e}^{-\mathrm{i}Ht}$ for a given Hamiltonian H. Moreover, Hamiltonian simulation serves as a crucial subroutine in many quantum algorithms. For example, quantum phase estimation,[9,11–13] Harrow–Hassidim–Lloyd algorithm,[1,15] and variational quantum eigensolver[16,17] algorithm utilize Hamiltonian simulation circuits or their variants.

Various Hamiltonian evolution algorithms[18–23] have been proposed. Currently, the simplest and most widely used algorithm is the Trotter–Suzuki decompositions.[19,21,24,25] This approach is easy to implement, and many experiments[26–28] of Hamiltonian simulation on quantum computers have adopted this type of algorithm. The fundamental idea of the algorithm is to divide the evolution time into many small segments and then repeatedly apply short-time evolution N_T times. The short-time evolution allows the Hamiltonian H to be decomposed and evolved sequentially.

The Hamiltonian simulation circuit depth of the 1st order Trotter–Suzuki decomposition scales as $\mathcal{O}(L^3)$ with the number of Hamiltonian terms L. If higher-order Trotter–Suzuki decompositions are used, the complexity with respect to L can be reduced to $\mathcal{O}(L^2)$ at best. For fermionic Hamiltonians in chemistry problems, such as electronic structure Hamiltonians, the number of terms scales as $\mathcal{O}(N^4)$. Consequently, the resulting scaling, from $\mathcal{O}(N^8)$ to $\mathcal{O}(N^{12})$, is technically efficient but prohibitively large in practice.

The quantum stochastic drift protocol (QDRIFT)[20] Hamiltonian simulation method addresses some of the challenges associated with Trotter–Suzuki decompositions. It achieves Hamiltonian evolution by randomly sampling terms from the Hamiltonian and encoding them into the simulation circuits, with the magnitudes of their coefficients serving as sampling weights. Refer to part II of Supplementary Materials (SM) for a detailed introduction. QDRIFT can reduce the circuit depth to $\mathcal{O}(L^2)$, and for many physical systems, even to $\mathcal{O}(L)$. Although there are other algorithms such as quantum signal processing[22] achieving better scaling in terms of simulation error, evolution time and other parameters, product formula-based methods like Trotter–Suzuki and QDRIFT often perform better in practice. Due to their simplicity and the fact that they do not require auxiliary qubits, they are also predominantly used in experimental implementations. As a result, theoretically optimal methods are often ill-suited for quantum computers in the noisy intermediate-scale quantum (NISQ)[29] era. Therefore, QDRIFT emerges as a promising candidate for implementation on NISQ devices.

*Corresponding authors. Email: qingchun720@ustc.edu.cn; wuyuchun@ustc.edu.cn
© 2025 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

2. Weighted Pauli Weight (WPW). Simulating fermionic systems using quantum simulation algorithms requires fermion-to-qubit mappings to transform the fermionic Hamiltonian into a Pauli Hamiltonian, which is a linear combination of multi-qubit Pauli operators. The earliest mapping is the Jordan–Wigner (JW) transformation.[30] Under this transformation, the Pauli weight of a multi-qubit Pauli operator corresponding to a creation or annihilation operator scales as $\mathcal{O}(N)$. Here, the Pauli weight refers to the number of non-trivial single-qubit Pauli operators in a multi-qubit Pauli operator. In Trotter–Suzuki decompositions, the circuit depth required to evolve a single Hamiltonian term is proportional to its Pauli weight. Consequently, many mappings[31–34] aim to reduce the Pauli weight to enhance the efficiency of the Hamiltonian simulation.

However, when using the QDRIFT simulation algorithm, a lower Pauli weight does not necessarily result in a shallower circuit. In QDRIFT, different Hamiltonian terms appear in the evolution circuit with frequencies proportional to the magnitudes of their coefficients. As a result, even a term with a large Pauli weight may contribute little to the overall circuit depth if its coefficient is small and it appears infrequently in the circuit.

Thus, in QDRIFT, both the Pauli weight of each term and the magnitude of its coefficient should be taken into account. To better characterize the circuit depth of Hamiltonian evolution under QDRIFT, we define a new metric, weighted Pauli weight (WPW), as

$$W_{\mathrm{w}}(H) = \sum_{l} |h_l| w(H_l), \tag{1}$$

where h_l is the coefficient of the Hamiltonian term H_l, and $w(H_l)$ denotes the Pauli weight of that term. The circuit depth in QDRIFT is proportional to W_{w} (see part II of the SM). The circuit depth of some other Hamiltonian simulation algorithms is also related to the WPW, such as the algorithm introduced by Nakaji *et al.*[35] A smaller circuit depth corresponds to shorter computation time and reduced noise-induced errors. Furthermore, we observe that reducing the WPW metric can also mitigate errors induced by readout noise. For a detailed discussion on how the WPW reduction suppresses the impact of noise on both Hamiltonian evolution and expectation value measurements, refer to part III of the SM. Therefore, our goal is to design fermion-to-qubit mappings that minimize the WPW.

3. Design of the Huffman-Code-Based Ternary Tree (HTT) Transformation. In 2019, Jiang *et al.* proposed the ternary tree (TT) transformation,[32] which represents the mapping using a balanced TT, i.e., a tree in which each non-leaf node has exactly three child nodes. Subsequent work[36] extended this transformation by allowing each node to have one to three child nodes. This generalized TT structure enables a broader class of mappings. Notably, well-known transformations such as the JW and Bravyi–Kitaev (BK) mappings can also be represented by corresponding TTs. For further details, see part IV of the SM.

We reformulate the task of designing a fermion-to-qubit mapping as the problem of constructing a generalized TT. Since the Hamiltonian coefficients must be taken into account, the optimal structure of the TT depends on the specific Hamiltonian under consideration. However, the number of possible TTs grows exponentially with the number of nodes (corresponding to the number of qubits), making exhaustive search computationally infeasible. To address this challenge, we draw inspiration from classical algorithms to efficiently construct the TT. This problem is closely analogous to designing optimal binary encodings for file storage in classical computing.

Suppose we want to store a text file on a computer that contains distinct characters, each of which must be encoded as a binary string. If all characters occur with equal frequency, then a fixed-length binary code is optimal, corresponding to a balanced binary tree. However, in real-world text, character frequencies vary, for instance, in English, the letter 'e' appears much more frequently than 'z'. To improve efficiency, shorter binary codewords are assigned to more frequent characters and longer codewords to less frequent ones. Huffman introduced the Huffman algorithm,[37] which constructs a generalized binary tree based on character frequencies. Once constructed, the tree yields an optimal code. For more details, refer to part V of the SM.

Within the framework of the TT transformation, we construct a new fermion-to-qubit mapping aimed at reducing the WPW. The construction procedure comprises two stages. First, a suitable generalized TT is generated, from which $2N$ Pauli strings are derived. Second, each of these Pauli strings is assigned to one of the $2N$ Majorana operators. It is not feasible to exhaustively search all configurations to find the configuration that minimizes the WPW for a given Hamiltonian. We extend the method used in the Huffman code algorithm, which can be generalized to construct a TT with low WPW according to the Hamiltonian. Moreover, since there exist numerous possible assignments between the generated Pauli strings and the Majorana operators, we further develop specialized allocation algorithms to identify an effective assignment strategy.

The Huffman code can be naturally extended to the ternary case. In the binary version, the two nodes with the lowest frequencies are merged to form a new parent node. Analogously, in the ternary extension, the three nodes with the smallest frequencies are combined to create a new node, which then serves as the parent of its three child nodes, designated as the left, middle, and right children.

We utilize the ternary Huffman algorithm to construct a generalized TT tailored to the coefficients of the Hamiltonian terms. In this framework, the $2N$ Majorana operators in the Hamiltonian are treated analogously to characters in a text file, with their associated frequencies determined

by the corresponding coefficients. Specifically, we define the frequency F_j of the Majorana operator m_j as

$$F_j = \sum_l I_l(j)|h_l|, \quad I_l(j) = \begin{cases} 1, \text{ if } m_j \in H_l, \\ 0, \text{ otherwise,} \end{cases} \tag{2}$$

where H_l denotes a term in the Hamiltonian, which is typically a product of two or four Majorana operators, and h_l is the corresponding coefficient. Once the set $\{F_j\}$ has been determined, the ternary Huffman algorithm is applied to construct the TT. During the merging process, each newly created node must be assigned an index corresponding to a qubit. Since the specific index assignment does not significantly affect the final mapping, indices are assigned in the reverse order of their creation. The complete procedure for generating the TT is described in Algorithm 1.

Algorithm 1: Generate the TT

Input: $\{m_j\}$ and $\{F_j\}$
Output: TT
TERNARY-HUFFMAN($\{F_j\}$);
$n = |\{m_j\}|$;
$Q = \{m_j\}$;
for $i = 1$ *to* n **do**
  $Q[j]$.freq $= F_j$;
  $Q[j]$.index $= j$;
for $i = 1$ *to* n **do**
  allocate a new node z;
  z.left=EXTRACT-MIN(Q);
  z.middle=EXTRACT-MIN(Q);
  z.right=EXTRACT-MIN(Q);
  z.freq $= z$.left.freq + z.middle.freq + z.right.freq;
  z.index $= 2N - i$;
  INSERT(Q, z);
return EXTRACT-MIN(Q);

After constructing the TT, we obtain $2N + 1$ paths from the root node to the leaf nodes, each corresponding to a unique Pauli string. The single-qubit Pauli operators in the Pauli strings are determined by the type of nodes. If the node is a left child node, the Pauli operator acting on the corresponding qubit is Pauli X; if the node is a middle child node, the Pauli operator is Pauli Y; and the Pauli operator is Pauli Z for a right child node. Among these $2N + 1$ Pauli strings, only $2N$ Pauli strings are independent. Removing one Pauli string from the set yields $2N$ independent Pauli strings.

The next step is to assign the $2N$ Majorana operators to these $2N$ Pauli strings. Without the requirement of satisfying Criterion (D) (see part IV of the SM), a straightforward strategy is to directly map the Pauli strings represented by paths from the root node to each leaf node to the Majorana operators represented by the leaf nodes. This method would ensure that the WPW of the Hamiltonian is minimized if each term in the Hamiltonian consists of only a single Majorana operator. However, in molecular Hamiltonians, the terms typically involve products of two or four Majorana operators. In such cases, minimizing the length of individual Pauli strings does not necessarily lead to the minimal length of Pauli strings for the operator products.

Our ultimate goal is to strive to ensure that the Pauli strings corresponding to Hamiltonian terms with larger coefficients are shorter in length. To achieve this, we allocate pairs of Majorana operators with strong interactions to Pauli strings such that their product yields a Pauli string of shorter length. Although the Hamiltonian contains both two-body and four-body terms, we treat each four-body term as a product of two two-body terms. Since a two-body term may appear in multiple Hamiltonian terms, directly using the coefficients of the Hamiltonian terms to represent interaction strength is not appropriate. We therefore redefine a two-body frequency to quantify the interaction strength. The frequency of a two-body term is defined as

$$F_{ij} = \sum_l I_l(ij)|h_l|, \tag{3}$$

where $I_l(ij)$ is a indicator defined as

$$I_l(ij) = \begin{cases} 1, \text{ if } m_i = H_l[1] \text{ and } m_j = H_l[2], \\ 1, \text{ if } m_i = H_l[3] \text{ and } m_j = H_l[4], \\ 0, \text{ otherwise,} \end{cases} \tag{4}$$

where $H_l[k]$ denotes the k-th Majorana operator in H_l. For instance, if $H_l = m_2 m_4 m_6 m_7$, then $H_l[2]$ is m_4. To avoid redundancy, we compute F_{ij} only for index pairs satisfying $i < j$. Furthermore, each term H_l is adjusted such that the Majorana operator with the smaller index precedes the one with the larger index. With the set $\{F_{ij}(i < j)\}$ computed, the mapping from the $2N$ Majorana operators to the $2N$ Pauli strings is then carried out using Algorithm 2.

Algorithm 2: Mapping Majorana operators to Pauli strings

Input: $M_{\text{set}} = \{m_j\}$, $S_{\text{set}} = \{S_k\}$, $F_{\text{set}} = \{F_{ij}\}$
Output: R_{dict}
$R_{\text{dict}} = \{\}$; `// R_dict is a dictionary`
Sort($Fset$); `// sort` $\{F_{ij}\}$ `in descending order`
while $Length(S_{set}) > 0$ **do**
  $F_{kl} = F_{\text{set}}[0]$;
  if m_k *in* R_{dict} *and* m_l *in* R_{dict} **then**
    continue ;
  else if m_k *in* R_{dict} **then**
    $S_k = R_{\text{dict}}[m_k]$;
    $Near_set = Find_near(S_{\text{set}}, S_k)$;
    $S_l = Find_short(Near_set)$;
    $R_{\text{dict}}[m_l] = S_l$;
    Delete S_l from S_{set};
  else if m_l *in* Map_dic **then**
    $S_l = R_{\text{dict}}[m_l]$;
    $Near_set = Find_near(S_{\text{set}}, S_l)$;
    $S_k = Find_short(Near_set)$;
    $R_{\text{dict}}[m_k] = S_k$;
    Delete S_k from S_{set};
  else
    $Near_pair_set = Find_near_pair(S_{\text{set}})$;
    $(S_k, S_l) = Find_short_pair(Near_pair_set)$;
    $R_{\text{dict}}[m_k] = S_k$;
    $R_{\text{dict}}[m_l] = S_l$;
    Delete S_k, S_l from S_{set}
return R_{dict};

The basic idea of the algorithm is to assign the pair of Majorana operators with the largest F_{ij} in $\{m_j\}$ to a pair of Pauli strings in $\{S_k\}$ whose product results in the shortest Pauli string among all possible pairwise combinations in $\{S_k\}$. If Criterion (D) is to be satisfied, each pair of Pauli strings must be assigned to the corresponding pair of Majorana operators m_{2j} and m_{2j+1}. For details on how the Pauli strings are paired, refer to part IV of the SM. Usually, there are multiple pairs of Pauli strings whose product yields the shortest resulting Pauli string. In such instances, we select the pair with the lowest total Pauli weight. Once a pair of Majorana operators and a corresponding pair of Pauli strings have been allocated, they are removed from the sets $\{m_j\}$ and $\{S_k\}$. Then we will allocate a new pair of Majorana operators with the largest F_{ij} in the new $\{m_j\}$ to a new pair of Pauli strings in the updated $\{S_k\}$. This procedure continues until all Majorana operators have been assigned.

If Criterion (D) is not enforced, the situation becomes more intricate, as a two-body term may involve one Majorana operator that has already been assigned and another that remains unassigned. In such cases, the unassigned operator is mapped to the Pauli string that is most similar to the Pauli string associated with the assigned Majorana operator. If there are multiple equally similar Pauli strings, the shortest one is selected. In most cases, the allocation process completes before all two-body terms have been iterated through. Consequently, two-body terms with larger coefficients are more likely to be assigned shorter resulting Pauli strings, whereas those with smaller coefficients may not fully satisfy this criterion. However, the impact of such low-coefficient terms on the new metric is limited. Although the allocation procedure does not guarantee a globally optimal value of the new metric, it effectively ensures that the metric attains a favorable value in practice.

4. Numerical Test. We compare the WPW for electronic structure Hamiltonians of various molecules under different mappings, and the results are presented in Table 1. In addition to the JW transformation, we also compare with the commonly used BK transformation [31] and the theoretically optimal TT transformation in terms of Pauli weight. The Hamiltonians are calculated using the ChemiQ [38] packages.

For systems with a small number of qubits, the JW mapping exhibits advantages. However, as the system size increases, the performance of the JW mapping deteriorates significantly compared to the other mappings. For molecules beyond Mg, our proposed Huffman-code-based ternary tree (HTT) mapping generally achieves superior results. We also evaluate the Pauli weight under different mappings, as shown in Table 2. The HTT transformation yields lower WPW while maintaining relatively small Pauli weight. In particular, for larger molecules, the Pauli weight of HTT is significantly smaller than that of the JW and BK transformations, and is only slightly larger than that of the optimal TT transformation.

Nevertheless, the extent of improvement varies considerably across different molecules. Notably, molecules such as SiO_2 and KOH exhibit significant enhancements, whereas others, including C_2H_6 and HNO_3, demonstrate more modest gains. Upon investigating the coefficients associated with various Majorana operators for these molecules, we discovered that molecules with minor improvements tend to have similar coefficients for their Majorana operators. This observation is intuitive. In the Huffman code scheme, characters are assigned binary strings of different lengths based on their frequencies. When there is significant variation in character frequencies, the compression is more effective. However, when the frequencies are similar or identical, the advantage of Huffman codes over fixed-length codes becomes negligible.

Similarly, when the coefficients F_j corresponding to different Majorana operators within a molecule exhibit low variance, the potential for improvement using the HTT mapping is limited. For example, in the case of C_2H_6, the largest F_j is 11.2 while the smallest is 8.3, indicating little difference. In contrast, for KOH, the largest F_j reaches 23.5 and the smallest is 6.6, representing a nearly threefold difference. As a result, KOH derives more substantial benefits from the HTT mapping, owing to the greater disparity in its F_j values.

5. Conclusion and Discussion. In this work, we propose a novel metric, termed the WPW, which more accurately characterizes the circuit depth in the QDRIFT

Table 1. The weight Pauli weight of electronic structure Hamiltonians for various molecules. All the Hamiltonians are calculated in the STO-3G basis.

Molecule	JW	BK-Tree	TT	HTT
LiH	31.07	39.59	40.29	38.14
H_2O	191.21	218.35	209.15	197.24
NH_3	232.76	248.02	250.86	229.95
Mg	429.41	480.72	501.25	404.142
N_2	364.18	413.39	397.44	386.10
SiO_2	2191.46	1949.63	1916.12	1690.77
CO_2	1156.15	1112.59	1033.71	1013.18
C_2H_6	1470.81	1168.22	1101.39	1074.46
KOH	2775.57	2375.83	2305.94	1937.19
Cl_2	3486.58	3338.30	3179.17	3019.60
HNO_3	4499.04	3523.98	3186.71	3173.24
KCl	4080.34	3206.55	3067.238	2562.63

Table 2. The Pauli weight of electronic structure Hamiltonians for various Molecules. All the Hamiltonians are calculated in the STO-3G basis.

Molecule	JW	BK-Tree	TT	HTT
LiH	3888	3370	3526	3556
H_2O	7764	6891	6789	6841
NH_3	30550	26349	26285	25548
Mg	29244	24936	24124	23990
N_2	28392	23628	21937	22008
SiO_2	475248	297217	264248	262986
CO_2	222070	148535	147495	152493
C_2H_6	763182	529187	584123	578452
KOH	919212	577841	511973	511142
Cl_2	552200	361573	327043	328998
HNO_3	2743700	1655987	1530792	1517010
KCl	2162416	1292863	1124348	1127177

algorithm. Building upon this metric, we introduce the Huffman-inspired TT transformation, derived from the principles of the Huffman algorithm. Numerical experiments demonstrate that, in most cases, the WPW achieved by HTT is significantly lower than that of other commonly used mappings.

While the Huffman code is guaranteed to have optimality, the presence of products of operators in the Hamiltonian complicates the situation. Under the new metric, HTT transformation might not necessarily be optimal. If one aims to search for the optimal TT configuration by exploring all configurations, the complexity of such a search is evidently at least $\mathcal{O}(2^N)$. Our approach provides results that are generally superior to those of other mappings, consuming relatively few resources. This makes the proposed solution a reasonable choice. However, exploring whether the optimal mapping can be found under reasonable resource consumption remains worthwhile. Further research is needed to determine in what kinds of systems HTT transformation yields more significant improvements.

While the Huffman algorithm guarantees optimality in classical coding scenarios, the presence of operator products in quantum Hamiltonians complicates the situation. Under the proposed WPW metric, the HTT transformation is not necessarily optimal. Exhaustively searching for the optimal TT configuration is generally infeasible. Nevertheless, our approach achieves consistently better performance than existing mappings while maintaining low computational overhead, making it a practical and effective solution. Exploring whether the optimal mapping can be identified within reasonable computational limits remains an open and worthwhile direction.

Acknowledgments. This work was supported by the National Key Research and Development Program of China (Grant No. 2024YFB4504101), the National Natural Science Foundation of China (Grant No. 22303022), and the Anhui Province Innovation Plan for Science and Technology (Grant No. 202423r06050002).

References

[1] Harrow A W, Hassidim A, and Lloyd S 2009 *Phys. Rev. Lett.* **103** 150502
[2] Childs A M, Kothari R, and Somma R D 2017 *SIAM J. Comput.* **46** 1920
[3] Ambainis A 2012 *Variable time amplitude amplification and quantum algorithms for linear algebra problems STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*, Feb 2012, Paris, France. pp. 636–647
[4] McArdle S, Endo S, Aspuru-Guzik A, Benjamin S C, and Yuan X 2020 *Rev. Mod. Phys.* **92** 015003
[5] Cao Y, Romero J, Olson J P, Degroote M, Johnson P D, Kieferová M, Kivlichan I D, Menke T, Peropadre B, Sawaya N P D, Sim S, Veis L, and Aspuru-Guzik A 2019 *Chem. Rev.* **119** 10856
[6] Kim Y, Eddins A, Anand S, Wei K X, van den Berg E, Rosenblatt S, Nayfeh H, Wu Y, Zaletel M, Temme K, and Kandala A 2023 *Nature* **618** 500
[7] Martinez E A, Muschik C A, Schindler P, Nigg D, Erhard A, Heyl M, Hauke P, Dalmonte M, Monz T, Zoller P, and Blatt R 2016 *Nature* **534** 516
[8] Arute F, Arya K, Babbush R *et al.* 2020 arXiv:2010.07965 [quant-ph]
[9] Babbush R, Wiebe N, McClean J, McClain J, Neven H, and Chan G K L 2018 *Phys. Rev. X* **8** 011044
[10] Reiher M, Wiebe N, Svore K M, Wecker D, and Troyer M 2017 *Proc. Natl. Acad. Sci. U.S.A.* **114** 7555
[11] Cao Y, Romero J, and Aspuru-Guzik A 2018 *IBM Journal of Research and Development* **62** 6:1
[12] Kassal I, Whiteld J D, Perdomo-Ortiz A, Yung M H, and Aspuru-Guzik A 2011 *Annu. Rev. Phys. Chem.* **62** 185
[13] Lu D, Xu B, Xu N, Li Z, Chen H, Peng X, Xu R, and Du J 2012 *Phys. Chem. Chem. Phys.* **14** 9411
[14] Aspuru-Guzik A, Lindh R, and Reiher M 2018 *ACS Central Sci.* **4** 144
[15] Babukhin D V 2023 *Phys. Rev. A* **107** 042408
[16] Peruzzo A, McClean J, Shadbolt P, Yung M H, Zhou X Q, Love P J, Aspuru-Guzik A, and O'Brien J L 2014 *Nat. Commun.* **5** 4213
[17] McClean J R, Romero J, Babbush R, and Aspuru-Guzik A 2016 *New J Phys.* **18** 023023
[18] Hadeld S and Papageorgiou A 2018 *New J Phys.* **20** 043003
[19] Suzuki M 1992 *Phys. Lett. A* **165** 387
[20] Campbell E 2019 *Phys. Rev. Lett.* **123** 070503
[21] Childs A M, Su Y, Tran M C, Wiebe N, and Zhu S 2021 *Phys. Rev. X* **11** 011020
[22] Low G H and Chuang I L 2017 *Phys. Rev. Lett.* **118** 010501
[23] Low G H and Chuang I L 2019 *Quantum* **3** 6
[24] Suzuki M 1990 *Phys. Lett. A* **146** 319
[25] Berry D W, Ahokas G, Cleve R, and Sanders B C 2007 *Commun. Math. Phys.* **270** 359
[26] Barends R, Lamata L, Kelly J *et al.* 2015 *Nat. Commun.* **6** 7654
[27] Lanyon B P, Hempel C, Nigg D *et al.* 2011 *Science* **334** 57
[28] Brown K R, Clark R J, and Chuang I L 2006 *Phys. Rev. Lett.* **97** 050504
[29] Preskill J 2018 *Quantum* **2** 79
[30] Jordan P and Wigner E 1928 *Zeitschrift fur Physik* **47** 631
[31] Bravyi S B and Kitaev A Y 2002 *Annals of Physics* **298** 210
[32] Jiang Z, Kalev A, Mruczkiewicz W, and Neven H 2020 *Quantum* **4** 276
[33] Li Q S, Liu H Y, Wang Q, Wu Y C, and Guo G P 2022 *J. Chem. Phys.* **157** 134104
[34] Harrison B, Chiew M, Necaise J, Projansky A, Strelchuk S, and Whiteld J D 2024 arXiv:2409.04348 [quant-ph]
[35] Nakaji K, Bagherimehrab M, and Aspuru-Guzik A 2024 *PRX Quantum* **5** 020330
[36] Miller A, Zimborás Z, Knecht S, Maniscalco S, and García-Pérez G 2023 *PRX Quantum* **4** 030314
[37] Human D A 1952 *Proceedings of the IRE* **40** 1098
[38] Wang Q, Liu H Y, Li Q S, Li Y, Chai Y, Gong Q, Wang H, Wu Y C, Han Y J, Guo G C, and Guo G P 2021 arXiv:2106.10162 [quant-ph]