Performance of the Quantum Approximate Optimization Algorithm on the Maximum Cut Problem

Gavin E. Crooks*
Rigetti Computing, 775 Heinz Avenue, Berkeley, CA 94710, USA
(Dated: November 21, 2018)

The Quantum Approximate Optimization Algorithm (QAOA) is a promising approach for programming a near-term gate-based hybrid quantum computer to find good approximate solutions of hard combinatorial problems. However, little is currently know about the capabilities of QAOA, or of the difficulty of the requisite parameters optimization. Here, we study the performance of QAOA on the MaxCut combinatorial optimization problem, optimizing the quantum circuits on a classical computer using automatic differentiation and stochastic gradient descent, using `QuantumFlow`, a quantum circuit simulator implemented with `TensorFlow`. We find that we can amortize the training cost by optimizing on batches of problems instances; that QAOA can exceed the performance of the classical polynomial time Goemans-Williamson algorithm with modest circuit depth, and that performance with fixed circuit depth is insensitive to problem size. Moreover, MaxCut QAOA can be efficiently implemented on a gate-based quantum computer with limited qubit connectivity, using a qubit swap network. These observations support the prospects that QAOA will be an effective method for solving interesting problems on near-term quantum computers.

PACS numbers: 03.67.Ac, 03.67.Lx

Introduction – The development of quantum computers continues apace, with the prospect of that intermediate scale machines that exhibit a clear advantage over classical computers will arrive presently [1]. However, our understanding of how to develop noise resilient algorithms that can run on near-term quantum resources remains limited. For a few computational problems, we know that a gate-based quantum computer can outperform any conceivable classical algorithm [2]. But for most problems we do not know how to implement effective algorithms on noisy, near term architectures with a modest number of qubits.

One promising approach to solving combinatorial optimization problems on near-term machines is the Quantum Approximate Optimization Algorithm (QAOA) [3–22]. This is a heuristic method, which can be thought of as a time-discretization of adiabatic quantum computing [23]. Like a number of near-term approaches to quantum computing [24–27], QAOA is a hybrid classical-quantum algorithm that combines quantum circuits, and classical optimization of those circuits. The objective is a functional of the quantum state, which in turn is a function parameterized by one and two-qubit gates whose character can be continuously varied. We perform classical optimization on these continuous gate parameters to generate distributions with significant support on the optimal solution. The use of a quantum resource allows us to prepare a distribution over an exponentially large sample space in a short sequence of gates, and the use of classical optimization expands the range of problems that we can explore, and may lead to circuits that are relatively robust to imperfections in the implementation of the quantum computer. Rather than attempt to program the quantum computer, we instead train the computer to perform the task at hand.

However, because QAOA is a heuristic algorithm it is difficult to provide general complexity theoretic guarantees about the performance of QAOA compared to classical algorithms [3, 4, 6, 11]; of the number of gates needed for effective implementations; nor of the difficulty of the requisite optimization step. In this paper, we use classical simulation with automatic differentiation [28] of parameters and stochastic gradient descent to explore these issues.

Maximum cut – In this paper we study the performance of QAOA on the maximum cut (MaxCut) combinatorial optimization problem: Given a graph $G = (V, E)$ with nodes V and edges E, find a subset $S \in V$ such that the number of edges between S and $S \setminus V$ is maximized. This problem can be reduced to that of finding the ground state of an antiferromagnetic Ising model. MaxCut is in the APX-complete complexity class: Finding an exact solution is NP-hard [29], but there are efficient polynomial time classical algorithms that find an approximate answer within some fixed multiplicative factor of the optimum [30]. For MaxCut the polynomial time Goemans-Williamson algorithm guarantees an approximation ratio of 0.8785 [31], which is optimal assuming the unique games conjecture [32]. It is NP-hard to approximate MaxCut better than $16/17 \approx 0.9412$ [33, 34].

QAOA MaxCut [3, 5, 8–10, 15–17] – To implement MaxCut on a quantum computer using QAOA, we encode the graph structure into a cost Hamiltonian which is diagonal in the computational basis, and for which any bit string gives an energy which is the negative of the

* gavin@rigetti.com

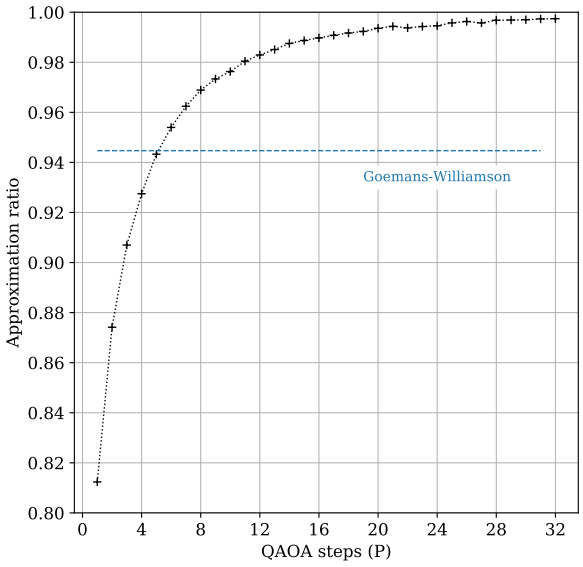


FIG. 1. The average approximation ratio of QAOA on the MaxCut optimization problem for 10 node graphs, as a function of QAOA steps (P). Each point represents an independently optimized protocol obtained via stochastic gradient descent. The training data consists of 100 graphs from the Erdös-Rényi ensemble with edge probability 50%, and the test data an independently sampled collection of 100 graphs from the same ensemble. For this problem size, QAOA with 5 steps matches the average performance of the classical, polynomial time Goemans-Williamson algorithm on the same data set.

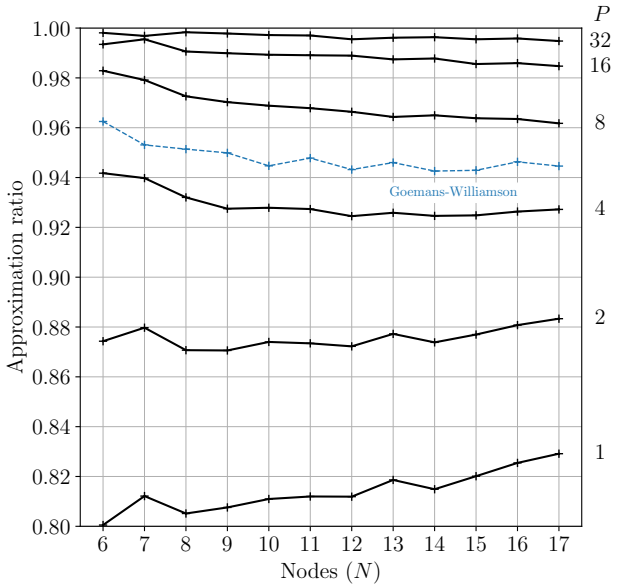


FIG. 2. The approximation ratio of QAOA on the MaxCut optimization problem, as a function of graph size (N) and QAOA steps (P). We also show the performance of the classical, polynomial time Goemans-Williamson algorithm on the same test sets. The performance of QAOA increases with circuit depth, and significantly exceeds that of Goemans-Williamson by $P = 8$. The performance of QAOA degrades as the graphs get bigger. However, quantum circuits with $P \geq 8$ maintain their relative performance advantage over the classical algorithm.

number of cut edges.

$$H_C = \frac{1}{2} \sum_{i,j \in E} C_{ij}(1 - \sigma_i^z \sigma_j^z) \tag{1}$$

Here σ_i^z is the Pauli Z matrix $\left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$ applied to qubit i, E is the set of edges, and C is the adjacency matrix of the graph, with $C_{ij} = 1$ if nodes are connected, and zero otherwise. (We can generalize this discussion to the problem of MaxCut on weighted graphs by substituting an edge weight matrix.)

We prepare the quantum computer in the state with uniform superposition of bit strings by applying a Hadamard gate to each qubit in the zero state. Then, for each of P steps of the QAOA algorithm, we evolve the system with the cost Hamiltonian for some angle γ_p, $U_p = \exp(-i\gamma_p H_C)$, and then evolve the system with a driver Hamiltonian

$$H_{\mathrm{D}} = \frac{1}{2} \sum_i \sigma_i^x \tag{2}$$

for an angle β_p, $V_p = \exp(-i\beta_p H_D)$, where σ_i^x is the Pauli X matrix $\left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$.

Repeated applications of the cost and driver dynamics evolves the system to the quantum state

$$|\boldsymbol{\beta}, \boldsymbol{\gamma}\rangle = V_P U_P \cdots V_2 U_2 V_1 U_1 |\psi\rangle \ . \tag{3}$$

Functional evaluation is performed by Monte-Carlo estimation of the cost Hamiltonian with samples drawn from the distribution given by $\rho(\boldsymbol{\beta}, \boldsymbol{\gamma}) = |\boldsymbol{\beta}, \boldsymbol{\gamma}\rangle\langle\boldsymbol{\beta}, \boldsymbol{\gamma}|$. Each quantum measurement provides access to a random variable h_C^i which is the cost Hamiltonian evaluated with the computational basis eigenstate resulting from a quantum measurement $|i\rangle$. To obtain an ϵ^2 variance on the estimator, $\widehat{\langle H_C \rangle}_{\rho(\boldsymbol{\beta},\boldsymbol{\gamma})}$, one can naively bound the number of samples, which are generated by a quantum state preparation and measurement, by applying the central limit theorem. Recall that the MaxCut problem can be encoded in a linear combination of 2-local Hamiltonians

$$H = \sum_{m=1}^{M} h_m P_m \tag{4}$$

where P_m is a tensor product of Z operators acting on at most 2-qubits and h_m is the strength of the interaction.

In the original QAOA algorithm the control parameters $(\boldsymbol{\beta}, \boldsymbol{\gamma})$ are optimized such that the functional, the expectation of the cost Hamiltonian for a given instance of the problem, $\langle\boldsymbol{\beta}, \boldsymbol{\gamma}|H_C|\boldsymbol{\beta}, \boldsymbol{\gamma}\rangle$, is minimized. However, this introduces a potentially expensive training step for every query. In the alternative, we train on batches of graphs drawn from the same statistical ensemble, and find a protocol $(\boldsymbol{\beta}^{\mathrm{opt}}, \boldsymbol{\gamma}^{\mathrm{opt}})$ that is effective for an entire

class of problem instances,

$$(\boldsymbol{\beta}^{\text{opt}}, \boldsymbol{\gamma}^{\text{opt}}) = -\arg\min_{\boldsymbol{\beta},\boldsymbol{\gamma}} \frac{1}{|\mathcal{T}|} \sum_{C \in \mathcal{T}} \langle \boldsymbol{\beta}, \boldsymbol{\gamma} | H_C | \boldsymbol{\beta}, \boldsymbol{\gamma} \rangle \ . \quad (5)$$

Here $\mathcal{T}$ in the training set of graph adjacency matrices, and $|\mathcal{T}|$ is the number of graphs in the training set. A similar approach was used by Wecker *et. al* [7].

We draw the initial QAOA parameters from a normal distribution. If the distribution of these initial parameters is overly broad, then the quantum circuits would be essentially random and hard to train, since the dynamics would be chaotic and information could not propagate through the network [35]. Similar issues occurs with deep neural networks [36, 37]. Empirically, we find that a standard deviation of 0.01 [38] and a mean of 0.5 (to avoid a symmetry about zero in the parameter space [16]) appears satisfactory.

Training – In order to explore quantum circuits for variational quantum algorithms, we implemented a simple quantum virtual machine (a simulation of a gate based quantum computer) on top of `TensorFlow` [39, 40]. This modern, high performance tensor processing library allows us to perform automatic differentiation of the performance metric with respect to the parameters of the quantum circuit. We can therefore train our quantum circuits using back-propagation and stochastic gradient descent. Code implementing our algorithm is available online [41].

Stochastic gradient descent has proved to be extremely effective in machine learning for training deep neural networks [37]. This makes SGD an attractive option for efficient exploration of quantum circuits using classical simulation due to accelerated optimization and ease of use [42]. Ideally, gradients would be calculated with respect to the entire dataset, but this is generally computationally expensive. Instead, at each step of the optimization we calculate the gradient with respect some random subsample of the full dataset.

However, it is difficult to implement stochastic gradient descent directly on a quantum computer, since the requisite gradients are expensive to measure [9], requiring many observations for each gradient component. A variety of approaches have been used to optimize QAOA circuits, including Nelder-Mead [9, 13], Monte-Carlo [7, 12], quasi-Newton [9, 21], gradient descent [16], and Bayesian [15] methods. It remains an important open question as to the most effective and efficient approach to training variational quantum algorithms on a quantum computer.

The validation sets for each graph size consists of 100 randomly generated graphs draw from the Erdös-Rényi ensemble (Edge probability 50%). Training was performed on an independently sampled collection of 100 graphs drawn from the same distribution. There is some chance of overlap between, and redundancy within, the test and validation sets. However, this will only be a significant issue for the smallest graphs, since the number of unique graphs grows rapidly with size, e.g. there are over

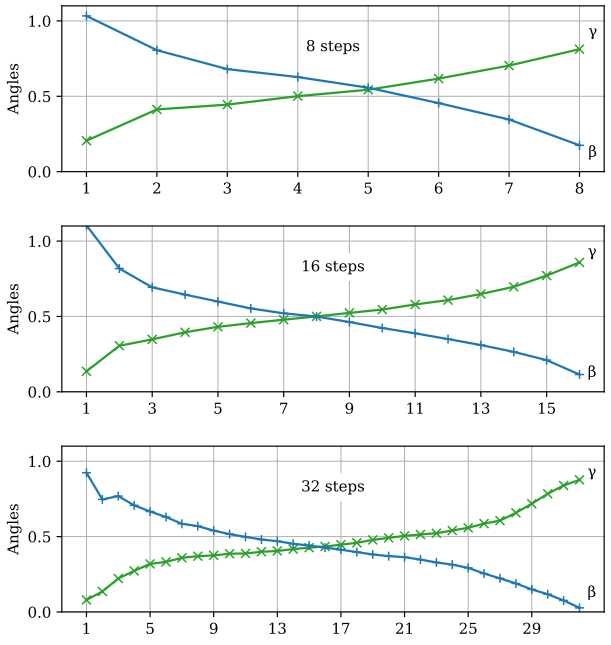


FIG. 3. Examples of optimized protocols $(\boldsymbol{\beta}^{\text{opt}}, \boldsymbol{\gamma}^{\text{opt}})$ for MaxCut QAOA on 10 node graphs, with 8, 16, and 32 QAOA steps. The resultant protocols are similar to a linear annealing schedule [23].

10^6 unique 10 node graphs. We use the Adam variant of stochastic gradient descent (which includes momentum and adaptive learning rates) [37, 43], with a mini-batch size of 1, a step size of 0.01, and other parameters at default. Both the γ and β parameters are periodic [3], but we do not constrain these values during optimization. Training from random initialization typically requires at most a few tens of epochs.

Performance – In Fig. 2 we show the approximation ratio as a function of graph size (N) and QAOA steps (P). Also shown is the average performance of the classical, polynomial time Goemans-Williamson algorithm on the same test sets. The performance of QAOA increases with circuit depth, and significantly exceeds that of Goemans-Williamson by $P = 8$. Although the approximation ratio does vary with problems size, circuits with $P \geq 8$ maintain their relative performance advantage over Goemans-Williamson. Notably there is no indication of a strong dependence of performance on the graph size for the Erdös-Rényi ensemble. Typical optimal protocols for different QAOA steps are illustrated in Fig. 3.

Computational resources – Since we can amortize the training cost, an analysis of the computation complexity of QAOA can focus on the number of gates needed to implement a single instance of the quantum evolution. And since the required number of QAOA steps (P) for a given performance does not appear to be strongly dependent on the problem size, the limiting resource is the number of two-qubit gates needed to implement a single round of

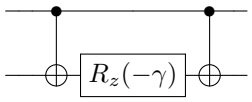
the algorithm. We focus on the number of 2-qubit gates, since we require at most two generic 1-qubit rotations per two-qubit gate.

Each of the Pauli-X interactions in the driver Hamiltonian (2) can be implemented with a single one-qubit gate,

$$e^{-\frac{i}{2}\beta\sigma_i^x} \equiv \quad \boxed{R_x(\beta)} \tag{6}$$

and each of the two-qubit ZZ interactions in the MaxCut cost Hamiltonian (1) can be implemented with two CNOT gates, plus a local one-qubit gate.

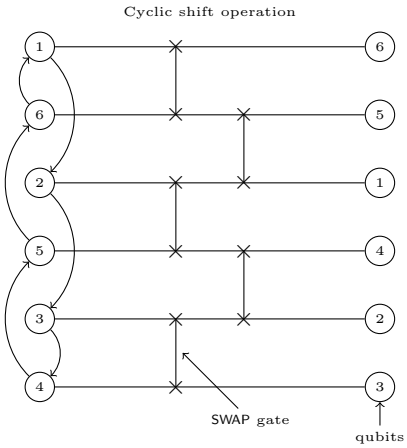
$$e^{-\frac{i}{2}\gamma(1-\sigma_i^z\sigma_j^z)} \equiv \tag{7}$$



Thus, for a fully connected graph of N nodes we require N qubits, and $N(N-1)P$ CNOTs.

These resource requirements assume that we can apply gates directly between any two qubits, but in practice qubits are typically arranged in a 2-dimensional lattice with gates only between nearest neighbors. SWAP gates can be used to move logical qubits into proximity, and naively one would expect that every logical gate would require $O(\sqrt{N})$ physical gates [44]. However, we can efficiently implement QAOA with a linear array of qubits using a SWAP network [45, 46], with $O(N)$ overhead.

Suppose we have a linear array of qubits, and we apply SWAP gates between all neighboring pairs in a brickwork pattern.



This gate architecture implements a ring buffer of qubits. Each application of the SWAP network rotates the qubits one position in the buffer. A full cycle of the buffer requires N applications of this cyclic shift operation, for a total of $N(N-1)$ SWAP gates.

The power of this architecture comes from noting that after half a cycle every qubit has been exchanged with every other qubit. At that point in the circuit, when any given pair of qubits are neighbors, we can also interleave the ZZ interactions required by the QAOA algorithm. This is possible because all of the ZZ interactions within a single QAOA iteration commute with one another, so the order that they are applied is unimportant. Each interaction requires 2 CNOTs [Eq. (7)], and a SWAP gate can be implemented with 3 CNOT gates. But we can combine the ZZ and SWAP gate into a 2-qubit parametric swap gate (PSWAP) [47]. And the PSWAP gate, as with any 2-qubit gate, can also be implemented with at most three CNOT gates [48–51]. Therefore the QAOA MAXCUT algorithm requires only $\frac{3}{2}N(N-1)P$ CNOT gates with a linear array of qubits. This is only a 50% overhead compared to the fully connected qubit topology.

Conclusions – Given that we can amortize the training cost; that we can exceed classical performance with a rather modest number of steps; and that QAOA can be efficiently implemented despite limited qubit connectivity, we expect QAOA MaxCut executed on a gate based quantum computer will require $O(N^2P)$ gates, and have a run time of $O(NP)$ (assuming we can apply $O(N)$ gates in parallel). At least for the small Erdös-Rényi graphs studied here, the requisite scaling of QAOA steps P with node number N appears to be sublinear. In contrast, Goemans-Williamson requires a running time of $\tilde{O}(Nm)$ for irregular graphs (ignoring logarithmic factors), where m is the number of edges [52].

Clearly, these observations are suggestive only, since it is prohibitively expensive to classically simulate the quantum MaxCut algorithm on anything but small graphs. A fair evaluation against state-of-the-art classical heuristic algorithms, such as simulated annealing and coherent Ising Machines [52, 53], would require graph problems with hundreds or thousands of nodes. Nonetheless, we are optimistic that QAOA can provide a significant quantum advantage [5, 17] on combinatorial optimization problems with modest numbers of qubits and modest gate depth. Definitive proof will have to await the anticipated arrival of a supremacy class quantum computer with sufficient gate fidelity [54, 55].

Acknowledgements – We heartily thank Matthew Harrigan, Jonathan Ward, Keri McKiernan, Chris Wilson, and Marcus da Silva for astute discussions. The circuit simulation framework of `QuantumFlow` is based upon Nicholas Rubin's `reference-qvm` [56], who also developed the Goemans-Williamson benchmark and contributed to the circuit depth analysis.

[1] J. Preskill, Quantum **2**, 79 (2018).

[2] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press,

2000).
[3] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," (2014), arXiv:1411.4028.
[4] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm applied to a bounded occurrence constraint problem," (2014), MIT-CTP/4628 arXiv:1412.6062.
[5] E. Farhi and A. W. Harrow, "Quantum supremacy through the Quantum Approximate Optimization Algorithm," (2016), arXiv:1602.07674.
[6] C. Y.-Y. Lin and Y. Zhu, "Performance of QAOA on typical instances of constraint satisfaction problems with bounded degree," (2016), MIT-CTP/4751 arXiv:1601.01744.
[7] D. Wecker, M. B. Hastings, and M. Troyer, Phys. Rev. A **94**, 022309 (2016).
[8] E. Farhi, J. Goldstone, S. Gutmann, and H. Neven, "Quantum algorithms for fixed qubit architectures," (2017), arXiv:1703.06199.
[9] G. G. Guerreschi and M. Smelyanskiy, "Practical optimization for hybrid quantum-classical algorithms," (2017), arXiv:1701.01450.
[10] S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, "From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz," (2017), arXiv:1709.03489.
[11] Z. Jiang, E. G. Rieffel, and Z. Wang, Phys. Rev. A **95**, 062317 (2017).
[12] Z.-C. Yang, A. Rahmani, A. Shabani, H. Neven, and C. Chamon, Phys. Rev. X **7** (2017).
[13] G. Verdon, M. Broughton, and J. Biamonte, "A quantum algorithm to train neural networks using low-depth circuits," (2017), arXiv:1712.05304.
[14] E. Zahedinejad and A. Zaribafiyan, "Combinatorial optimization on gate model quantum computers: A survey," (2017), arXiv:1708.05294.
[15] J. S. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. S. Fried, S. Hong, P. Karalekas, C. B. Osborn, A. Papageorge, E. C. Peterson, G. Prawiroatmodjo, N. C. Rubin, C. A. Ryan, D. Scarabelli, M. Scheer, E. A. Sete, P. Sivarajah, R. S. Smith, A. Staley, N. Tezak, W. J. Zeng, A. Hudson, B. R. Johnson, M. Reagor, M. P. da Silva, and C. T. Rigetti, "Unsupervised machine learning on a hybrid quantum computer," (2017), arXiv:1712.05771.
[16] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel, Phys. Rev. A **97**, 022304 (2018).
[17] A. M. Dalzell, A. W. Harrow, D. E. Koh, and R. L. La Placa, "How many qubits are needed for quantum computational supremacy?" (2018), arXiv:1805.05224.
[18] W. Lechner, "Quantum approximate optimization with parallelizable gates," (2018), arXiv:1802.01157.
[19] E. F. Dumitrescu, A. L. Fisher, T. D. Goodrich, T. S. Humble, B. D. Sullivan, and A. L. Wright, "Benchmarking treewidth as a practical component of tensor-network–based quantum simulation," (2018), arXiv:1807.04599.
[20] W. W. Ho, C. Jonay, and T. H. Hsieh, "Ultrafast state preparation via the Quantum Approximate Optimization Algorithm with long range interactions," (2018), arXiv:1810.04817.
[21] E. R. Anschuetz, J. P. Olson, A. Aspuru-Guzik, and Y. Cao, "Variational quantum factoring," (2018), arXiv:1808.08927.
[22] M. Fingerhuth, T. Babej, and C. Ing, "A quantum alternating operator ansatz with hard and soft constraints for lattice protein folding," (2018), arXiv:1810.13411.
[23] T. Albash and D. A. Lidar, Rev. Mod. Phys. **90**, 015002 (2018).
[24] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, Nat. Commun **5**, 4213 (2014).
[25] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, New J. Phys **18**, 023023 (2016).
[26] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature **549**, 242 (2017).
[27] I. H. Kim and B. Swingle, "Robust entanglement renormalization on a noisy quantum computer," (2017), arXiv:1711.07500.
[28] T. Tamayo-Mendoza, C. Kreisbeck, R. Lindh, and A. Aspuru-Guzik, "Automatic differentiation in quantum chemistry with an application to fully variational hartree-fock," (2017), arXiv:1711.08127.
[29] R. M. Karp, "Reducibility among combinatorial problems," (Springer US, Boston, MA, 1972) pp. 85–103.
[30] C. H. Papadimitriou and M. Yannakakis, J. Comput. Syst. Sci. **43**, 425 (1991).
[31] M. X. Goemans and D. P. Williamson, J. ACM **42**, 1115 (1995).
[32] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell, SIAM J. Comput. **37**, 319 (2007).
[33] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, J. ACM **45**, 501 (1998).
[34] J. Håstad, J. ACM **48**, 798 (2001).
[35] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," (2018), arXiv:1803.11173.
[36] X. Glorot and Y. Bengio, in *Proceedings of Machine Learning Research*, Vol. 9, edited by Y. W. Teh and M. Titterington (PMLR, 2010) pp. 249–256.
[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, Massachusetts, 2016).
[38] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," (Springer, Berlin, 2012) pp. 599–619, 2nd ed.
[39] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)* (2016) pp. 265–283.
[40] `QuantumFlow`: A Quantum Algorithms Development Toolkit `https://quantumflow.readthedocs.io/`.
[41] `https://github.com/rigetticomputing/quantumflow-qaoa`.
[42] D. Sels, Phys. Rev. A **97**, 040302 (2018).
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," (2014), arXiv:1412.6980.
[44] D. Cheung, D. Maslov, and S. Severini, "Translation techniques between quantum circuit architectures," (2007), workshop on Quantum Information Processing.
[45] I. D. Kivlichan, J. McClean, N. Wiebe, C. Gidney, A. Aspuru-Guzik, G. K.-L. Chan, and R. Babbush, Phys. Rev. Lett. **120**, 110501 (2018).

[46] R. Babbush, N. Wiebe, J. McClean, J. McClain, H. Neven, and G. K.-L. Chan, Phys. Rev. X **8**, 011044 (2018).
[47] R. S. Smith, M. J. Curtis, and W. J. Zeng, "A practical quantum instruction set architecture," (2016), arXiv:1608.03355.
[48] F. Vatan and C. Williams, Phys. Rev. A **69**, 032315 (2004).
[49] G. Vidal and C. M. Dawson, Phys. Rev. A **69**, 010301 (2004).
[50] J. Zhang, J. Vala, S. Sastry, and K. B. Whaley, Phys. Rev. A **69**, 042309 (2004).
[51] V. V. Shende, I. L. Markov, and S. S. Bullock, Phys. Rev. A **69**, 062321 (2004).
[52] Y. Haribara, S. Utsunomiya, K.-i. Kawarabayashi, and Y. Yamamoto, "Principles and methods of quantum information technologies," (Springer, Tokyo, 2016) Chap. A coherent Ising machine for MAX-CUT problems: Performance evaluation against semidefinite programming and simulated annealing.
[53] R. Hamerly, T. Inagaki, P. L. McMahon, D. Venturelli, A. Marandi, T. Onodera, E. Ng, C. Langrock, K. Inaba, T. Honjo, K. Enbutsu, T. Umeki, R. Kasahara, S. Utsunomiya, S. Kako, K.-i. Kawarabayashi, R. L. Byer, M. M. Fejer, H. Mabuchi, E. Rieffel, H. Takesue, and Y. Yamamoto, "Scaling advantages of all-to-all connectivity in physical annealers: the Coherent Ising Machine vs. D-Wave 2000Q," (2018), arXiv:1805.05217.
[54] J. Preskill, "Quantum computing and the entanglement frontier," (2012), arXiv:1203.5813.
[55] M. Reagor, C. B. Osborn, N. Tezak, A. Staley, G. Prawiroatmodjo, M. Scheer, N. Alidoust, E. A. Sete, N. Didier, M. P. da Silva, E. Acala, J. Angeles, A. Bestwick, M. Block, B. Bloom, A. Bradley, C. Bui, S. Caldwell, L. Capelluto, R. Chilcott, J. Cordova, G. Crossman, M. Curtis, S. Deshpande, T. El Bouayadi, D. Girshovich, S. Hong, A. Hudson, P. Karalekas, K. Kuang, M. Lenihan, R. Manenti, T. Manning, J. Marshall, Y. Mohan, W. O'Brien, J. S. Otterbach, A. Papageorge, J.-P. Paquette, M. Pelstring, A. Polloreno, V. Rawat, C. A. Ryan, R. Renzas, N. Rubin, D. Russel, M. Rust, D. Scarabelli, M. Selvanayagam, R. Sinclair, R. Smith, M. Suska, T.-W. To, M. Vahidpour, N. Vodrahalli, T. Whyland, K. Yadav, W. Zeng, and C. T. Rigetti, Science Advances **4**, eaao3603 (2018).
[56] `https://github.com/rigetticomputing/reference-qvm`.