Latency considerations for stochastic optimizers in variational quantum algorithms

Matt Menickelly[1], Yunsoo Ha[2], and Matthew Otten[3]

[1]Mathematics and Computer Science Division, Argonne National Laboratory, 9700 S. Cass Ave., Lemont, IL 60439

[2]Edward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, 915 Partners Way, Raleigh, NC 27601

[3]HRL Laboratories, LLC, 3011 Malibu Canyon Road, Malibu, CA 90265

Variational quantum algorithms, which have risen to prominence in the noisy intermediate-scale quantum setting, require the implementation of a stochastic optimizer on classical hardware. To date, most research has employed algorithms based on the stochastic gradient iteration as the stochastic classical optimizer. In this work we propose instead using stochastic optimization algorithms that yield stochastic processes emulating the dynamics of classical deterministic algorithms. This approach results in methods with theoretically superior worst-case iteration complexities, at the expense of greater per-iteration sample (shot) complexities. We investigate this trade-off both theoretically and empirically and conclude that preferences for a choice of stochastic optimizer should explicitly depend on a function of both latency and shot execution times.

1 Introduction

Quantum computers have the potential to perform many important calculations faster than their classical counterparts do. Disparate domains such as quantum chemistry [1]; nuclear [2], condensed matter [3], and high energy [4] physics; machine learning and data science [5]; and finance [6] are all theorized to benefit from quantum algorithms in various ways. In the near-term, the so-called noisy intermediate-scale quantum (NISQ) [7] era, these theoretical benefits are hard to realize, because the canonical quantum algorithms that are used in many of the domains, such as quantum phase estimation [8], require gate depths that are expected to be realized only with fault-tolerant, error-corrected quantum computers [9]. Variational quantum algorithms (VQAs) attempt to lower gate depth requirements by replacing some of the requirements with the need to perform optimization using classical computers [10]. Such algorithms have shown success on NISQ hardware in eigenvalue estimation [11], dynamical evolution [12, 13, 14], machine learning [15, 16], and many other problems [10]. The variation quantum eigensolver (VQE) is perhaps the best-known example, with many demonstrations performed on a variety of physical and chemical systems [14, 17]. One of the key bottlenecks in VQAs is the optimization step; the functions being optimized, as well as their higher-order derivative information, need to be estimated via a limited number of samples, necessitating the use of stochastic optimization algorithms. A simple way of quantifying the total cost would be through the number of shots: for each function evaluation, the quantum computer needs to be queried many times to get an estimate of the function to be used in a classical optimization routine. These sampling queries are in addition to the need to run many different sets of parameters along the optimization trajectory, which introduces a "circuit switching" latency. Given that many of today's quantum computers are available in a cloud setting, there can be additional overhead due to network latency [18]. These latencies can also vary among different architectures; for example, a superconducting transmon quantum processor has measurement times on the order of a few microseconds [19], while a trapped ion system can take hundreds of microseconds [20, 21]. This is, of course, in addition to gate times and reset times, all of which increase the time to take a single sample. Designing optimization algorithms that take these various times into account is important for minimizing the total amount of potentially expensive quantum computer time used.

Matt Menickelly: mmenickelly@anl.gov
Yunsoo Ha: yha3@ncsu.edu
Matthew Otten: mjotten@hrl.com

A few stochastic gradient methods have been developed specifically for the quantum regime that attempt to minimize the number of shots [22], as well as those that attempt to minimize total wall time [18]. However, stochastic gradient methods, including the popular Adam optimizer [23], are typically deployed in applications where accuracy does not matter much. In quantum chemistry, where VQE is used extensively, there is a standard level of accuracy, known as "chemical accuracy," that is desirable because it is the minimum accuracy required to allow for chemically relevant predictions [24]. Many stochastic gradient methods also require careful hyperparameter tuning [25, 26], particularly in the selection of a step-size parameter. In VQAs, each sample (or observation) can be expensive because quantum computer time is highly limited. Doing extensive hyperparameter sweeps is, therefore, untenable.

1.1 Our Contributions

Given these circumstances, we suggest a new optimization algorithm, SHOt-Adaptive Line Search (SHOALS). In each iteration, SHOALS computes a stochastic estimate of the gradient of the cost function, computes a trial step, and evaluates the cost function at both the current set of parameters and the trial step. If sufficient decrease within a confidence interval is detected, the step is accepted, and the trial step becomes the current set of parameters for the next iteration. In every iteration, SHOALS updates certain accuracy parameters that, when coupled with conservative concentration inequalities, determine sample sizes for computing function and gradient values.

The convergence theory behind SHOALS is derived from ALOE [27], which in turn is an extension of previous work in [28, 29, 30]. These works in adaptive optimization consider the quantities involved in deterministic methods of optimization (in particular, trust region methods and line search methods) and replace these quantities with random variables to yield an analyzable stochastic process. As a result, one derives stochastic variants of deterministic methods that yield, up to constants, the worst-case iteration complexity guarantees of the deterministic methods with high probability; SHOALS achieves this by mimicking the dynamics of gradient descent with a line search. These results are significant because the iteration complexities of deterministic methods are of a different order than those of stochastic methods. For instance, in the worst case, given Lipschitz continuous f, the number of iterations that a deterministic gradient descent method requires to attain $\|\nabla f(\theta^k)\| \leq \epsilon$ is on the order of $1/\epsilon^2$; this bound is known to be tight [31, 32].[1] In the same class of problems, the number of iterations for a first-order method based on the classical Robbins-Monro stochastic gradient iteration to achieve the same degree of stationarity is on the order of $1/\epsilon^4$ [36], which is strictly worse.[2] In the quantum computing setting, adopting an algorithm like SHOALS translates to significantly lower total time, since certain large parts of the latency costs are paid per iteration, rather than per shot, at the expense of potentially higher (but dynamic) numbers of shots. We note that a recent paper on another classical optimizer for VQAs employs a Bayesian line search [39], which is fundamentally different from the line search considered in this paper.

We stress that SHOALS is not a new algorithm, but is a practical extension of ALOE [27]. Specifically, SHOALS allows for per-parameter sampling of directional derivatives, enabled by parameter shift gradients. However, we do note that this is the first work to explicitly consider adaptive stochastic methods for use in VQAs, and we illustrate - in terms of latency cost and shot execution tradeoffs - why such algorithms are extremely promising in this setting, even though they were not explicitly designed for this setting.

We provide numerical experiments demonstrating the efficacy of SHOALS on a variety of chemical molecules. Depending on the specifics of a simulated quantum computing environment, SHOALS can reduce the time needed to reach chemical accuracy by several orders of magnitude compared with other state-of-the-art optimization algorithms, such as iCANS [22] and Adam. In particular, our comparisons employ various measures of latency, showing how the effect of circuit switching and network latency costs affect the comparisons. As suggested by the theory on which we elaborate in this paper, we find that SHOALS frequently outperforms pop-

[1]In general, the worst-case iteration complexity for *any* unconstrained deterministic first-order optimization method to attain $\|\nabla f(\theta^k)\| \leq \epsilon$ given a Lipschitz continuous gradient and Hessian is on the order of $\epsilon^{-1.75}$ [33]. Such a theoretically best worst-case rate is indeed attained by accelerated gradient descent [34, 35].

[2]In general, the theoretical best worst-case iteration complexity for *any* unconstrained stochastic first-order optimization method to attain $\|\nabla f(\theta^k)\| \leq \epsilon$ - the definition of which must be more carefully provided than in this footnote - is on the order of $1/\epsilon^3$ [37]. Such a theoretical best rate is attained, for instance, by SPIDER [38], at the cost of evaluating two stochastic gradients per iteration (as opposed to one stochastic gradient per iteration required by the classical Robbins-Monro iteration).

ular methods based on the stochastic gradient iteration, especially in settings where latency cost is high compared with shot cost. Our algorithm and numerical experiments highlight the importance of making more efficient use of limited quantum resources, allowing for further exploration of more interesting problems.

1.2 Variational Quantum Eigensolver

In the VQE, a standard example of a VQA [10], we seek to find an approximation of the lowest eigenvalue of a Hermitian matrix, H, which is often known as the Hamiltonian. VQE relies on what is known as the "variational principle," which states

$$\langle\psi(\theta)|H|\psi(\theta)\rangle \geq E_0, \tag{1}$$

where E_0 is the true value of the lowest eigenvalue (often known as the ground state energy) and $|\psi(\theta)\rangle$ is any parameterizable function (typically known as an ansatz). The variational principle, Eq. (1), states that an upper bound for the true ground state energy can be obtained by measuring the energy (that is, computing $\langle\psi(\theta)|H|\psi(\theta)\rangle$) for any function, $|\psi(\theta)\rangle$. A flexible, expressive wavefunction $|\psi(\theta)\rangle$ and efficient, effective optimization are key to realizing high-quality approximations of the true ground state energy. Because of their natural ability to express quantum mechanics, quantum computers are excellent candidates for preparing various ansatzes.

For standard quantum chemistry problems, we can write the Hamiltonian, H, after preprocessing through a fermion-to-spin transformation such as the Jordan–Wigner transformation [40], as a sum of terms,

$$H = \sum_{j=0}^{N} a_j P_j, \tag{2}$$

where a_j is a (typically real-valued) constant prefactor and P_j is a Pauli string. Note that the N here is often expressed as $\mathcal{O}(N_o^4)$, where N_o is the number of orbitals used (and is often equal to the number of qubits). Each Pauli string can be measured efficiently on a quantum computer. The VQE problem can thus be expressed as an optimization problem,

$$\min_\theta \sum_j^N a_j \langle\psi(\theta)|P_j|\psi(\theta)\rangle. \tag{3}$$

Each of the terms in the sum in (3) is independently estimated stochastically by repeatedly querying the quantum computer for some number of shots.

For convenience of notation in discussing an optimization algorithm, we will, in the remainder of this paper, abuse notation and rewrite (3) as

$$\min_\theta \sum_j^N a_j P_j(\theta) \equiv \min_\theta f(\theta), \tag{4}$$

where $P_j(\theta)$ denotes $\langle\psi(\theta)|P_j|\psi(\theta)\rangle$. By using the parameter shift rule [41], partial derivatives for the types of ansatzes used in this work can be calculated as

$$\partial_i f(\theta) = \frac{f(\theta + \frac{\pi}{2}e_i) - f(\theta - \frac{\pi}{2}e_i)}{2}, \tag{5}$$

where e_i represents the unit vector in the direction of parameters i. We utilize a Trotterized, unitary coupled cluster, singles doubles (UCCSD) ansatz [42, 43] for this work, which is defined as

$$|\psi(\theta)\rangle = \left(\prod_i^d \prod_j \exp \theta_{i,j}\hat{t}_j\right)|0\rangle, \tag{6}$$

where d represents what we term depth, which is the number of times to repeat the basic Trotterized UCCSD ansatz with unique parameters (similar to the k parameter in Ref. [42]), and $\hat{t}_j$ is either a singles ($\hat{t}_j = \hat{a}_k^\dagger \hat{a}_l$) or doubles ($\hat{t}_j = \hat{a}_k^\dagger \hat{a}_l^\dagger \hat{a}_m \hat{a}_n$) cluster operator. We note that our theory should extend to any ansatz, such as those generated in qubit coupled cluster [44] or hardware-efficient ansatzes [14], and can be embedded within methods to generate more compact ansatzes, such as ADAPT-VQE [45] and unitary selective coupled cluster [46].

1.3 Single-Shot Estimators

A gradient-based optimization algorithm for solving (4) will require the ability to compute $f(\theta)$ and $\nabla f(\theta)$. However, because each term $P_j(\theta)$ in (4) is in fact a random variable for which a shot execution on a quantum computer gives an estimate valued in $\{-1, 1\}$, we cannot directly compute $f(\theta)$. Instead, we obtain what we call a *single-shot estimator* $f(\theta; \xi)$ of $f(\theta)$, which is computed by taking one observation of each Pauli string $P_j(\theta)$ and returning the linear combination specified by the prefactors $\{a_j\}$. We note that although we call $f(\theta; \xi)$ a *single-shot* estimator, it does not cost one shot to obtain a single-shot estimator; rather, a single-shot estimator costs one shot per circuit necessary to evaluate every Pauli string, which is typically $O(N_q^4)$, where N_q is the number of qubits used. Using operator grouping, one potentially can reduce this number to $O(N_q^3)$ [47]. The random variable ξ in $f(\theta; \xi)$ is intended to encapsulate all the randomness inherent in estimating $f(\theta)$ with a quantum computer.

Analogously, using the parameter shift gradient, Eq. (5), we can compute a single-shot estimator $g_i(\theta;\xi)$ of each partial derivative $\partial_i f(\theta)$. When we assemble the single-shot estimators of partial derivatives into a full gradient vector, we denote this latter quantity $g(\theta;\xi)$.

2 Shot-Adaptive Line Search

Our method is fundamentally based on a stochastic line search method proposed over several papers [29, 30] but is closest to one referred to as Adaptive Linesearch with Oracle Estimations, or ALOE [27]. To disambiguate from the use of the word "oracle" in quantum computing, we instead use the name SHOt-Adaptive Line Search, or SHOALS. We describe this method in Algorithm 1 and discuss key elements of the algorithm.

Algorithm 1: SHOt-Adaptive Line Search (SHOALS)

```
1  **Initialization:** Choose constants
     $\gamma > 1, c \in (0,1)$, and $\alpha_{\max} > 0$.
2  Choose accuracy level $\epsilon_f > 0$.
3  **Input:** initial point $\theta^0 \in \mathbb{R}^n$, initial step size
     $\alpha_0 > 0$.
4  **for** $k = 0,1,2,\ldots$ **do**
5      Compute an estimate $g^k$ of $\nabla f(\theta^k)$ (in
         practice; $g^k$ will be assembled
         coordinate-wise from averages of $N_{g_i,k}$
         samples of $g_i(\theta^k;\xi)$; see (7)).
6      Set a trial point $\mathbf{s}^k \leftarrow \theta^k - \alpha_k g^k$.
7      Compute function estimates $f_0^k$ and $f_{\mathbf{s}}^k$ of
         $f(\theta^k)$ and $f(\mathbf{s}^k)$, respectively (in
         practice, $f_0^k$ and $f_s^k$ will be computed as
         averages of $N_{f,k}$ samples of $f(\theta^k;\xi)$ and
         $f(\mathbf{s}^k;\xi)$; see (9)).
8      **if** $f_{\mathbf{s}}^k \le f_0^k - c\alpha_k \|g^k\|^2 + 2\epsilon_f$ **then**
9          $\theta^{k+1} \leftarrow \mathbf{s}^k$
10         $\alpha_{k+1} \leftarrow \min\{\alpha_{\max}, \gamma\alpha_k\}$
11     **else**
12         $\theta^{k+1} \leftarrow \theta^k$
13         $\alpha_{k+1} \leftarrow \gamma^{-1}\alpha_k$
```

We note that Algorithm 1 would be a classical line search method with an Armijo line search with the notable differences that

- Line 10 would be replaced with $\alpha_k \leftarrow 1$;
- the estimates g^k, f_0^k, and f_s^k would be replaced with the respective true values $\nabla f(\theta^k), f(\theta^k)$, and $f(\mathbf{s}^k)$; and
- ϵ_f would always be initialized to 0.

Under particular conditions on the estimates g^k, f_0^k, and f_s^k, as well as the determination of ϵ_f in Line 7, Jin et al. [27] demonstrate a special kind of convergence result concerning the iterates of Algorithm 1. We now elaborate on these conditions, which will be enforced in SHOALS.

2.1 Gradient Estimation

In our setting, a directional derivative estimate $g_i^k \approx \partial_i f(\theta^k)$ (see (5)) is computed by averaging a number $N_{g_i,k}$ of single-shot estimates of the directional derivative, $g_i^k(\theta^k;\xi)$. As a result, the gradient estimate g^k can be written $g^k(\theta^k;\xi^k)$, where ξ^k denotes a realization of the random variable Ξ^k representing all possible shot sequences that could be observed when obtaining $N_{g_i,k}$ many single-shot estimates of each $\partial_i f(\theta^k)$, and concatenating the estimates into a n-dimensional gradient vector. We stress that in our notation, whenever ξ has a superscript (k), there is an implied (set of) sample size(s) $N_{g_i,k}$ to distinguish it from the atomic random variable ξ involved in the single-shot estimator.

We give reasoning in the appendix, but we ultimately require, for each $i = 1,\ldots,n$

$$N_{g_i,k} = \left\lceil \frac{\mathbb{V}\left[g_i(\theta^k;\xi)\right]}{p\left(\max\{L_i\alpha_k g_i(\theta^k;\xi^k), \epsilon_g\}\right)^2} \right\rceil, \qquad (7)$$

for some $p \in (0, \frac{1}{2}^n)$ and some $\epsilon_g > 0$. In (7), L_i denotes a global Lipschitz constant of $\partial_i f(\theta)$ and $\mathbb{V}\left[g_i(\theta^k;\xi)\right]$ denotes the variance of the single-shot estimator $g_i(\theta^k;\xi)$. Loosely speaking, enforcing (7) will guarantee that, with high probability $(1-p)$, the error in the estimate $g_i(\theta^k;\xi^k)$ of $\partial_i f(\theta^k)$ will be bounded in such a way that that it does not interfere with the gradient descent dynamics.

As in [22], we note that we can yield an upper bound on each L_i due to the parameter shift rule and the expression of the cost function as a prefactor-weighted sum of Pauli matrices. In particular, given an expression for the non-mixed partial second derivative $\partial_i\partial_i f(\theta^k)$, we can yield an upper bound on each L_i by the sum of the absolute values of prefactors obtained after applying a parameter shift (5) twice.

To make practical use of (7), it remains to handle the unknown variance term in the numerator. The variance of the estimator $g_i(\theta^k;\xi)$ is unknowable because computing a population variance requires knowing the expectation $\nabla f(\theta^k)$. In our implementation we simply compute the sample variance of the sample of size $N_{g,k}$ before the end of the

kth iteration to use as an estimate of the variance in the $(k+1)$st iteration. It is theoretically reasonable to believe that the variance $\mathbb{V}\left[g(\theta^k;\xi)\right]$ should not change too rapidly in θ^k, justifying this choice. We have also verified this assumption empirically through experiments. To summarize, our practical sample size is

$$N_{g_i,k} = \left\lceil \frac{s^2_{g_i,k}}{p\left(\max\{L_i\alpha_k g_i(\theta^k;\xi^k), \epsilon_g\}\right)^2} \right\rceil, \tag{8}$$

where $s^2_{g_i,k}$ is the sample variance estimate of $g_i(\theta^k;\xi)$ obtained in the previous iteration.

2.2 Function Value Estimation

Similarly to the gradient estimation, we average a number $N_{f,k}$ of single-shot estimates of $f(\theta^k)$ and $f(\mathbf{s}^k)$ to yield estimates $f(\theta^k;\xi^k)$ and $f(\mathbf{s}^k;\xi^k)$.

$$N_{f,k} = \left\lceil \frac{\mathbb{V}\left[f(\theta^k;\xi)\right]}{p\epsilon_f^2} \right\rceil, \tag{9}$$

where $\epsilon_f > 0$ is a parameter, and $p \in (\frac{1}{2}, 1)$. Formal reasoning is given in the appendix, but intuitively, averaging $N_{f,k}$ many samples guarantees with high probability that the error incurred by $f(\theta^k;\xi^k)$ is bounded by a constant ϵ_f that dictates the accuracy to which we would intend to reduce the optimality gap of our final solution.

In our practical implementation, to avoid oversampling, we additionally employ a criterion resembling past work and the criterion in (8) (see [48, 29]) and compute

$$N_{f,k} = \min\left\{ \left\lceil \frac{s^2_{f,k}}{p(\alpha_k^2\|g(\theta^k;\xi^k)\|^2)^2} \right\rceil, \left\lceil \frac{s^2_{f,k}}{\epsilon_f^2} \right\rceil \right\}, \tag{10}$$

where $s^2_{f,k}$ is the sample variance estimate of $\mathbb{V}[f(\theta^k;\xi)]$, which would have been computed in the previous iteration.

2.3 Theoretical Results concerning SHOALS

In [27], a more formal statement of the following result is proven.

Theorem 2.1. *Suppose f is bounded below. that is, there exists $f_* = f(\theta^*)$ so that $f_* \le f(\theta)$ for all θ, and suppose $\nabla f(\theta)$ is Lipschitz continuous with constant L_g. Fix $\epsilon_g > 0$ and $\epsilon_f > 0$. Suppose the estimators $g(\theta^k;\xi^k)$ and $\{f(\theta^k;\xi^k), f(\mathbf{s}^k;\xi^k)\}$ employed in each iteration of Algorithm 1 satisfy Condition A.1 and Condition A.2 with ϵ_g and ϵ_f, respectively. Let*

$$\epsilon \ge 4\max\{\epsilon_g, (1+L_g\alpha_{\max})\sqrt{3L_g\epsilon_f}\}.$$

Then, with high probability, the number of iterations of Algorithm 1, T_ϵ, required to attain

$$\|\nabla f(\theta^{T_\epsilon})\| < \epsilon \tag{11}$$

satisfies

$$T_\epsilon \in \mathcal{O}\left(\frac{L_g^2\big(f(\theta^0) - f(\theta^*)\big)}{\epsilon^2}\right). \tag{12}$$

The probability in this result is with respect to the σ-algebra of the realizations of ξ^k observed over the run of the algorithm.

We remark on some important aspects of Theorem 2.1. First, as indicated by the big-$\mathcal{O}$ notation, this result provides a *worst-case* guarantee. Virtually never in practice is a stopping time on the order of $1/\epsilon^2$ many iterations realized. However, the result of Theorem 2.1 recovers up to small constants the worst-case iteration complexity result for *deterministic* gradient descent algorithms, which is known to be tight [31].

2.4 Comparison with Stochastic Gradient Methods

The result in Theorem 2.1 should be contrasted with known results for the worst-case performance of stochastic gradient methods. We provide a prototypical stochastic gradient method in Algorithm 2. Results in Corollary 2.2 of Ref. [36] and Theorem 4.8

Algorithm 2: Generic Stochastic Gradient Method

```
1 Initialization: Choose step size $\alpha$, initial point $\theta^0 \in \mathbb{R}^n$
2 for $k = 0, 1, 2, \ldots$ do
3     Compute an estimate $g^k$ of $\nabla f(\theta^k)$.
4     $\theta^{k+1} \leftarrow \theta^k - \alpha g^k$.
```

of Ref. [49] show essentially that in order to guarantee

$$\mathbb{E}\left[\frac{1}{T_\epsilon}\sum_{k=0}^{T_\epsilon}\|\nabla f(\theta^k)\|\right] \le \epsilon, \tag{13}$$

one must run a stochastic gradient method for

$$T_\epsilon \in \mathcal{O}\left(\frac{L_g(f(\theta^0) - f(\theta^*))}{\epsilon^4}\right) \tag{14}$$

many iterations, provided the step size α is chosen sufficiently small, typically satisfying a bound like $\alpha \le \frac{1}{L_g}$. For ease of presentation, we deliberately ignored in (13) and (14) an additional dependence on the variance of g^k, which decreases linearly with the sample size or "batch size" b. This is a significant oversight because in the worst case one requires $b \in \Theta(T_\epsilon)$, where T_ϵ is as in (14) in order for theoretical results such as (13) to hold. This is certainly concerning in settings where accuracy matters (e.g., chemical accuracy in the problems we are considering), but in many machine learning applications it is considerably less concerning. In light of this intentional oversight, we remark that the worst-case results we are presenting for SG methods are, in fact, erring on the side of optimism.

The bound in (13) can be interpreted as saying that, given a fixed number of iterations of T, the *expected value* (over the σ-algebra of all sources of randomness) of the average value (averaged over $k = 1, \ldots, T$) of $\|\nabla f(\theta^k)\|$ is bounded by ϵ. As a corollary, one can show that if one selects uniformly at random $R \in \{1, \ldots, T_\epsilon\}$, then

$$\mathbb{E}\left[\|\nabla f(\theta^R)\|\right] \leq \epsilon. \tag{15}$$

These results have been shown to be tight in the general case [50] and improvable to $\mathcal{O}\left(\frac{1}{\epsilon^{3.5}}\right)$ in special cases [51]. Popular variants of SG algorithms like Algorithm 2, such as Adam, have been shown to have virtually the same type and quality of error bounds as that in (13); see Theorem 1 of Ref. [52]. Methods such as iCANS [22] are also based on the stochastic gradient iteration and thus are susceptible to the same worst-case performance.

The trade-off between a $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ worst-case complexity for algorithms of the class in Algorithm 1 and the $\mathcal{O}\left(\frac{1}{\epsilon^4}\right)$ worst-case complexity for algorithms of the class in Algorithm 2 is that the estimates in Algorithm 1 require more samples (or, in the context of our parameter shift problems, shots) per iteration to satisfy Condition A.1 and Condition A.2. The appropriate metric for comparing these algorithms is measured by the *total work*, namely, a function of latency and shot acquisition times.

We consider the following simplified setting. We note that similar analyses (see [53] and [49, Section 4.4]) have been done to explain the apparent empirical dominance of stochastic gradient methods in empirical risk minimization settings, but the conclusions we will draw are very different because the quantum computing setting is very different from general machine learning settings in terms of overhead costs. We denote the shot acquisition time by c_1, the latency time of circuit switching by c_2, and the communication cost time by c_3. All times vary from architecture to architecture. Shot acquisition time includes the time to reset the circuit to the initial state, the time to run the circuit, and the time to record a single measurement, which can from a around 100 μs in superconducting qubits [54] and around 200 ms in neutral atom systems [55]. Circuit switching time includes the time needed to compile a circuit and then to load it on to the quantum computer's control system. For superconducting qubits, compilation can take 200 ms, while interacting with the arbitrary waveform generator can take around 25 ms [54]. Communication cost includes the time needed to communicate between the computer running the classical optimization algorithm and the actual quantum computer. This can range from nearly zero when the two devices are in the same room to many seconds when instructions need to be sent over the Internet [18]. We note that every iteration of Algorithm 1 requires two communications between the quantum and classical devices: one communication to obtain $g(\theta^k;\xi^k)$ and a second communication to obtain estimates $f(\theta^k;\xi^k)$ and $f(\mathbf{s}^k,\xi^k)$. Every iteration of Algorithm 2 requires only one communication to obtain $g(\theta^k;\xi^k)$. Thus, denoting by $t_\circ^f$ and $t_\circ^g$ the number of circuits required to evaluate the one-shot estimators $f(\theta^k;\xi)$ and $g(\theta^k;\xi)$, respectively, we can summarize the overhead *per-iteration latency cost* of each iteration as in Table 1.

Now, denote by t_s^f and t_s^g the number of shots requested per iteration. By Theorem 2.1, if we choose $\epsilon = \epsilon_g < 1$ and $\epsilon_f = \epsilon^2$, then in the worst case, and up to constants, Algorithm 1 will require on each iteration $\frac{1}{\epsilon^2}$ samples of $g(\theta^k;\xi)$ and $\frac{1}{\epsilon^4}$ samples of $f(\theta^k;\xi)$, yielding $t_s^g \leq \frac{t_\circ^g}{\epsilon^2}$ and $t_s^f \leq \frac{2t_\circ^f}{\epsilon^4}$. On the other hand, each iteration of Algorithm 2 will require only a constant number of samples, b, of $g(\theta^k;\xi)$. That is, without loss of generality, and up to constants, $t_s^g = bt_\circ^g$. All of these costs are summarized in Table 1.

We note that SHOALS incurs only the communication cost, c_3, on the order of $1/\epsilon^2$ many times, whereas the cost is incurred on the order of $1/\epsilon^4$ many times for SG methods. Additionally, the latency switching cost c_2 is present only on the $1/\epsilon^2$ term for SHOALS, whereas it appears on the order of $1/\epsilon^4$ many times in SG methods. On the other hand, we note that SHOALS has an unfortunate $1/\epsilon^6$ dependence; however, if c_1 is small—that is, if the cost of shot acquisition is small relative to the total latency and communication costs $c_3 + c_2(t_\circ^f + t_\circ^g)$)—then the $1/\epsilon^6$ term may not dominate.

To quantify this trade-off a bit more, we make simplifying assumptions that $t_\circ^g = 2nt_\circ^f$, where n is the dimension of the parameter vector θ, which is likely correct up to constants in the parameter

	Iterations	$\times$ (Per-Iteration Latency Cost	+ Per-Iteration Shots Cost)
SHOALS	$\mathcal{O}\left(\frac{L_g^2(f(\theta^0)-f(\theta^*))}{\epsilon^2}\right)$	$c_2(2t_\circ^f+t_\circ^g)+2c_3$	$c_1\left(\frac{t_\circ^g}{\epsilon^2}+\frac{2t_\circ^f}{\epsilon^4}\right)$
=	$\mathcal{O}\left(\frac{L_g^2(f(\theta^0)-f(\theta^*))\left[c_2(2t_\circ^f+t_\circ^g)+2c_3\right]}{\epsilon^2}+\frac{L_g^2(f(\theta^0)-f(\theta^*))c_1t_\circ^g}{\epsilon^4}+\frac{2c_1t_\circ^fL_g^2(f(\theta^0)-f(\theta^*))}{\epsilon^6}\right)$		
SG	$\mathcal{O}\left(\frac{L_g(f(\theta^0)-f(\theta^*))}{\epsilon^4}\right)$	$c_2t_\circ^g+c_3$	$bc_1t_\circ^g$
=	$\mathcal{O}\left(\frac{\left[bc_1t_\circ^g+c_2t_\circ^g+c_3\right]L_g(f(\theta^0)-f(\theta^*))}{\epsilon^4}\right)$		

Table 1: Summary of total worst-case costs of SHOALS and a stochastic gradient (SG) method to achieve ϵ-stationarity. The total worst-case cost is computed as the number of iterations to achieve ϵ-stationarity multiplied by the sum of the per iteration costs. We recall that $t_\circ^f$ and $t_\circ^g$ denote the number of circuits needed to compute $f(\theta^k;\xi)$ and $g(\theta^k;\xi)$, respectively. For an SG method, we denote a constant batch size parameter by b.

shift setting [41], and that the batchsize employed by the SG method is $b=1$, which is correct up to constants in practice. Then, in this setting and per Table 1, one can derive that a sufficient condition for the total cost of SHOALS being less than the total cost of an SG method is the satisfaction of

$$c_1 < \frac{\epsilon^2(2nt_\circ^f c_2 + c_3)}{2t_\circ^f L_g}, \tag{16}$$

still assuming without loss of generality that $L_g > 1$. That is, roughly speaking, if the cost of obtaining a single shot is smaller by a factor of ϵ^2 than the overhead latency cost of one iteration of an SG method, normalized by both the number of circuits needed to evaluate $f(\theta^k;\xi)$ and the gradient Lipschitz constant, then SHOALS is expected to yield better worst-case guarantees in terms of total costs than an SG method does. Thinking to the future of the deployment of VQAs, as n becomes large, L_g ought to increase linearly in N in (4) and hence linearly in n; therefore, (16) asympotically amounts to $c_1 < \epsilon^2 c_2$.

In our experiments we will examine this trade-off empirically; but to summarize, these theoretical insights suggest that, asymptotically, when shot acquisition time is less than a factor of ϵ^2 times the latency switching time, we expect SHOALS to outperform methods based on the iteration in Algorithm 2.

3 Numerical Results

We note that, similar to stochastic gradient methods, SHOALS is remarkably simple to implement. It has also been documented in past work that adaptive methods like SHOALS are far less sensitive to changes in hyperparameters (such as step sizes) than stochastic gradient methods are, virtually eliminating the need for tuning hyperparameters. In our implementation of SHOALS, we chose $\gamma=2, c=0.2, \alpha_{\max}=1$, and $\alpha_0=1$.

In our implementation, when computing sample sizes via (8) and (10) we set $p=0.1$. Motivated by Theorem 2.1, we ignore constant terms and set $\epsilon_g=\sqrt{\epsilon_f}$, where in turn motivated by chemical accuracy [24], we set $\epsilon_f = 0.0016$. We note that because $\epsilon \in \Theta(\epsilon_g)$ per Theorem 2.1, this essentially means that the factor of ϵ^2 that played a critical role in the end of our discussion in Section 2.4 is effectively ϵ_f.

We first consider a hypothetical, forward-looking, superconducting hardware environment outlined in Ref. [18]; in particular, we set $c_1 = 1.0\times10^{-5}$ s, $c_2 = 0.1$ s, and $c_3 = 4.0$ s. Before showing any results, we remark that in this setting, $c_1 < \epsilon_f c_2$; and so, from our discussion in Section 2.4, the theory suggests that in worst-case settings and for larger n, we expect SHOALS to outperform all of the stochastic gradient methods. Using these hardware times, we will display in our figures the total simulated time on the x-axis, where t_s^f and t_s^g (the numbers of shots requested by a method per iteration) are the actual values requested in the simulations.

We consider three quantum chemistry problems, for which we give relevant statistics in Table 2. For all molecules we use parity fermion-to-spin mapping with two-qubit reduction [56] to reduce the number of qubits, as implemented in Qiskit [57]. We simulate H_2 with two qubits at an equilibrium distance of 0.74 $\mathring{A}$. For LiH, we use an equilibrium distance of 1.595 $\mathring{A}$, and we freeze core orbitals (as in [14]), leading to only four qubits. We use a square geometry for H_4, with each side having equilibrium length 1.23 $\mathring{A}$ [42], and comprising six qubits.

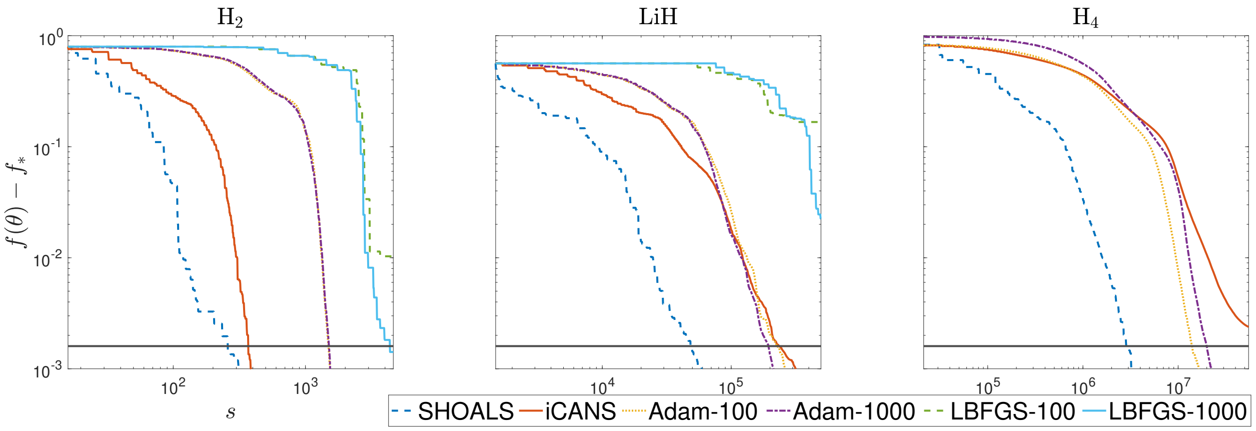

Figure 1: Comparison of SHOALS, iCANS, Adam, and LBFGS. Numbers next to Adam and LBFGS denote the shot size employed for each circuit. The x-axis shows the cost of the optimization in time in the simulated superconducting environment, while the y-axis shows the distance of the value of $f(\theta)$ to the known global minimum f_* after the corresponding number of budget units is spent on an optimizer. The solid horizontal line denotes chemical accuracy. For each solver, we show median performance over 30 runs with randomly generated initial points θ^0 (common to all solvers) with the line indicated in the legend. We remark again that LBFGS made negligible progress on H$_4$, and hence it is omitted from the rightmost figure.

Ansatz	n	$t_\circ^f$	$t_\circ^g$
H$_2$, UCCSD, Depth 1	3	5	40
LiH, UCCSD, Depth 2	16	104	8320
H$_4$, UCCSD, Depth 2	70	85	106080

Table 2: Relevant statistics for tested problems. As in preceding sections, n denotes the number of parameters in the ansatz, $t_\circ^f$ denotes the number of circuits that must be evaluated to obtain an evalution of $f(\theta^k;\xi)$, and $t_\circ^g$ denotes the number of circuits that must be evaluated to obtain an evaluation of $g(\theta^k;\xi)$.

3.1 Comparison of SHOALS with Other Solvers

We first compare the performance of SHOALS with two stochastic gradient methods: an implementation of iCANS and an implementation of Adam. For iCANS, we used the same hyperparameter settings described in [22], with a minimum sample size set equal to 30, and implemented what the authors referred to as "iCANS1." For Adam, we employed a constant step size of $1.0/L_g$, momentum hyperparameters of 0.9 and 0.999, and a denominator offset hyperparameter of 10^{-8}. These settings are standard in practice. We experimented with a batch size (in the quantum setting, average of a number of single-shot estimators $g(\theta^k;\xi)$) of both 100 and 1000. We remark that one could tune all of the hyperparameters of both iCANS and Adam to yield variations in practical performance, but one intention of these experiments is to demonstrate that *standard* settings of SHOALS' hyperparameters yield relatively good performance compared with *standard* settings of other solvers. Furthermore, we remark again that in actual NISQ settings, one would not want to expend budgets performing hyperparameter tuning.

Additionally, we compare the performance of SHOALS with a deterministic first-order method, an implementation of L-BFGS [58]. This implementation is the standard one provided in Scipy and Qiskit and was *not* designed to handle stochasticity in the function evaluation. Therefore, such a comparison is not particularly fair, since an L-BFGS optimizer has no guarantee of first-order convergence in such a setting. We experiment with L-BFGS employing both a batch size of 100 and 1000. We remark that there is ongoing work on the development of L-BFGS optimizers capable of handling stochastic estimates of function and gradient values [59, 60]. In fact, Pacquette and Scheinberg [29] discuss how to incorporate approximate second-order information, such as that provided by L-BFGS techniques, into the adaptive line search framework We leave the incorporation of such second-order information into SHOALS as future work, since it is not the focus of this current paper.

Results are illustrated in Figure 1, and additional statistics are given in Table 3. We see that, in this superconducting setting, the median wall-clock time performance of SHOALS is consistently better than any of the compared stochastic gradient and L-BFGS methods. As expected, on all of the tested

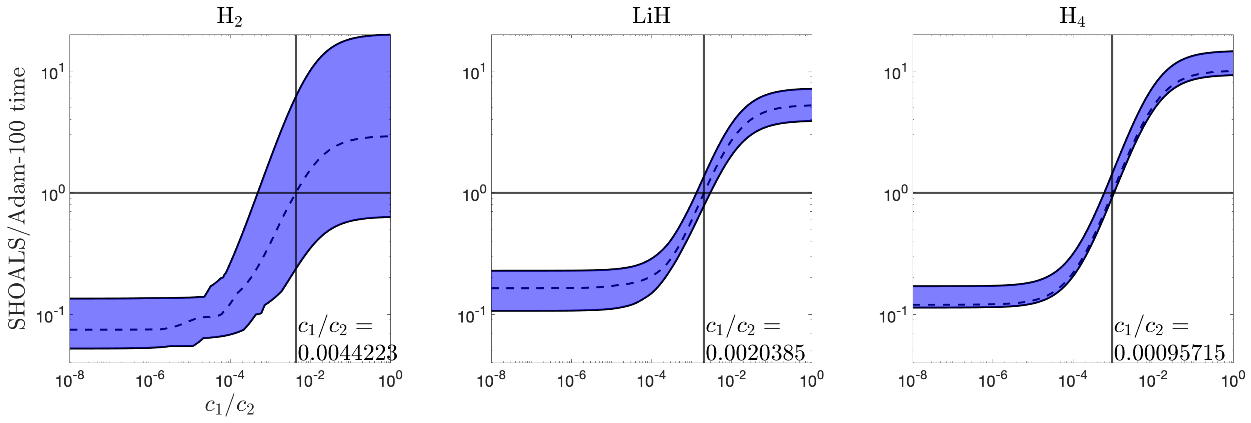

Figure 2: For each of the tested molecules, we display the ratio of the wall-clock time needed for SHOALS to achieve chemical accuracy over the wall-clock time needed for Adam-100 to achieve chemical accuracy. The dashed line denotes the median value of the ratio, while the transparent band shows the 25th through 75th percentiles of the ratio. We also display in each plot the value of c_1/c_2 where the median ratio equals 1, that is, where the wall-clock time of the two solvers is equal.

problems, the median performance of L-BFGS with a fixed batchsize fails to attain chemical accuracy. For this reason, we chose not to present that solver for the largest tested molecule, H_4.

In light of our discussion in Section 2.4 and the inherent uncertainty in the specific timings of the superconducting setting, we additionally present in Figure 2, for the same molecules, the median (and first and third quantiles) of the ratio of the wall-clock time needed for SHOALS to attain chemical accuracy to the wall-clock time needed for Adam-100 to attain chemical accuracy when we fix $c_3 = 0$ (i.e., we completely disregard cloud communication time) and vary the ratio of c_1/c_2.

As expected by our discussion in Section 2.4, for each tested molecules, as c_1/c_2 tends to 0, the advantage of using SHOALS over a stochastic gradient method is clear. What merits further investigation, but requires testing significantly larger molecules, is whether the trend that the value of c_1/c_2 at the breakeven point is decreasing with n. One explanation would be that our predictions are based on *worst-case* behavior and the optimization problems we are testing are certainly not exhibiting worst-case behavior. In any case, it would be foolhardy to extrapolate any trend from three datapoints; testing molecules that require orders of magnitudes more circuits to compute a single gradient will involve an extensive computational campaign, which is a subject for future work.

4 Conclusions and Future Work

In this paper we demonstrate a new optimization algorithm, SHOt-Adaptive Line Search (SHOALS). We provide theoretical arguments and numerical evidence that SHOALS should outperform other state-of-the-art optimization algorithms when the shot acquisition time is much less than the circuit switching time. This is because the dynamics of the deterministic line-search method that SHOALS stochastically approximates requires fewer iterations but more shots per iteration, to reach a given level of stationarity. Standard stochastic gradient methods, on the other hand, can utilize smaller numbers of shots per iteration to make expected improvement in the cost function. The latency regime in which SHOALS is expected to perform better than stochastic gradient methods is realized by today's superconducting qubit quantum computers [18]. However, when the circuit switching time and shot acquisition time are closer in magnitude, such as in trapped ion systems [20], we show, theoretically and numerically, that stochastic gradient methods are expected to outperform SHOALS. Because quantum devices have a non-negligible per-iteration overhead (from the need to switch between many circuits), these latency considerations are important in determining which algorithm to use in variational quantum algorithms, whereas in optimization of classical functions, such as those encountered in machine learning, these considerations are considerably less important.

In future work we intend to investigate the possibility of tighter worst-case complexities for

SHOALS, in terms of the wall-clock time metric, than those presented in Table 1. Indeed, as a simplifying assumption, we assumed that SHOALS required $\mathcal{O}(1/\epsilon^2)$ many samples of $g(\theta^k;\xi)$ and $\mathcal{O}(1/\epsilon^4)$ many samples of $f(\theta^k;\xi)$, $f(\mathbf{s}^k;\xi)$ per iteration, but this is in fact necessary only when k is near the stopping time T_ϵ. A more rigorous analysis would consider the *total work* of SHOALS, such as in the analysis performed in Ref. [61].

We also intend to pursue multiple directions in the practical implementation of SHOALS. In one direction, we are interested in the incorporation of (approximate) second-order information into SHOALS in order to further decrease the iteration complexity and hence relax further the total accumulated per-iteration latency switching costs. As suggested previously, we are inspired by work in Refs. [59, 60, 29]. We also intend to investigate an operator-sampling variant of SHOALS, in the style of Ref. [62], in order to further reduce both latency and shot acquisition costs. In particular, we may choose on certain iterations, based on sample variance estimates, to evaluate only a subset of Pauli strings in (4), or perhaps to evaluate only a subset of directional derivative estimates $g_i(\theta^k;\xi^k)$ when forming a gradient estimate g^k. Such approaches would resemble doubly stochastic coordinate descent methods, which have been studied in the optimization literature [63].

Moreover, and as suggested in our numerical experiments, we intend to conduct a larger computational campaign to look at larger molecules in order to ascertain our belief in the asymptotic performance of SHOALS.

5 Acknowledgements

This work was supported in part by the U.S. Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research Applied Mathematics activity and AIDE-QC project under contract number DE-AC02-06CH11357.

References

[1] Benjamin P Lanyon, James D Whitfield, Geoff G Gillett, Michael E Goggin, Marcelo P Almeida, Ivan Kassal, Jacob D Biamonte, Masoud Mohseni, Ben J Powell, Marco Barbieri, et al. "Towards quantum chemistry on a quantum computer". Nature Chemistry **2**, 106–111 (2010).

[2] Ian C Cloët, Matthew R Dietrich, John Arrington, Alexei Bazavov, Michael Bishof, Adam Freese, Alexey V Gorshkov, Anna Grassellino, Kawtar Hafidi, Zubin Jacob, et al. "Opportunities for nuclear physics & quantum information science" (2019). arXiv:1903.05453.

[3] Adam Smith, MS Kim, Frank Pollmann, and Johannes Knolle. "Simulating quantum many-body dynamics on a current digital quantum computer". npj Quantum Information **5**, 1–13 (2019).

[4] Benjamin Nachman, Davide Provasoli, Wibe A de Jong, and Christian W Bauer. "Quantum algorithm for high energy physics simulations". Physical Review Letters **126**, 062001 (2021).

[5] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. "Quantum machine learning". Nature **549**, 195–202 (2017).

[6] Roman Orus, Samuel Mugel, and Enrique Lizaso. "Quantum computing for finance: Overview and prospects". Reviews in Physics **4**, 100028 (2019).

[7] John Preskill. "Quantum computing in the NISQ era and beyond". Quantum **2**, 79 (2018).

[8] U Dorner, R Demkowicz-Dobrzanski, BJ Smith, JS Lundeen, W Wasilewski, K Banaszek, and IA Walmsley. "Optimal quantum phase estimation". Physical Review Letters **102**, 040403 (2009).

[9] John Preskill. "Fault-tolerant quantum computation". In Introduction to quantum computation and information. Pages 213–269. World Scientific (1998).

[10] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. "Variational quantum algorithms". Nature Reviews PhysicsPages 1–20 (2021).

[11] Peter JJ O'Malley, Ryan Babbush, Ian D Kivlichan, Jonathan Romero, Jarrod R McClean, Rami Barends, Julian Kelly, Pedram Roushan, Andrew Tranter, Nan Ding, et al. "Scalable quantum simulation of molecular energies". Physical Review X **6**, 031007 (2016).

[12] Xiao Yuan, Suguru Endo, Qi Zhao, Ying Li, and Simon C Benjamin. "Theory of variational quantum simulation". Quantum **3**, 191 (2019).

[13] Matthew Otten, Cristian L Cortes, and Stephen K Gray. "Noise-resilient quantum dynamics using symmetry-preserving ansatzes" (2019). arXiv:1910.06284.

[14] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M Chow, and Jay M Gambetta. "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets". Nature **549**, 242–246 (2017).

[15] Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. "Quantum circuit learning". Physical Review A **98**, 032309 (2018).

[16] Matthew Otten, Imène R Goumiri, Benjamin W Priest, George F Chapline, and Michael D Schneider. "Quantum machine learning using gaussian processes with performant quantum kernels" (2020). arXiv:2004.11280.

[17] Robert M Parrish, Edward G Hohenstein, Peter L McMahon, and Todd J Martínez. "Quantum computation of electronic transitions using a variational quantum eigensolver". Physical Review Letters **122**, 230401 (2019).

[18] Kevin J Sung, Jiahao Yao, Matthew P Harrigan, Nicholas C Rubin, Zhang Jiang, Lin Lin, Ryan Babbush, and Jarrod R McClean. "Using models to improve optimizers for variational quantum algorithms". Quantum Science and Technology **5**, 044008 (2020).

[19] Jay Gambetta, WA Braff, A Wallraff, SM Girvin, and RJ Schoelkopf. "Protocols for optimal readout of qubits using a continuous quantum nondemolition measurement". Physical Review A **76**, 012325 (2007).

[20] Susan M Clark, Daniel Lobser, Melissa C Revelle, Christopher G Yale, David Bossert, Ashlyn D Burch, Matthew N Chow, Craig W Hogle, Megan Ivory, Jessica Pehr, et al. "Engineering the quantum scientific computing open user testbed". IEEE Transactions on Quantum Engineering **2**, 1–32 (2021).

[21] Colin D Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M Sage. "Trapped-ion quantum computing: Progress and challenges". Applied Physics Reviews **6**, 021314 (2019).

[22] Jonas M Kübler, Andrew Arrasmith, Lukasz Cincio, and Patrick J Coles. "An adaptive optimizer for measurement-frugal variational algorithms". Quantum **4**, 263 (2020).

[23] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization" (2014). arXiv:1412.6980.

[24] Trygve Helgaker, Poul Jorgensen, and Jeppe Olsen. "Molecular electronic-structure theory". John Wiley & Sons. (2014).

[25] Tom Schaul, Ioannis Antonoglou, and David Silver. "Unit tests for stochastic optimization". In Yoshua Bengio and Yann LeCun, editors, 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings. (2014). url: `http://arxiv.org/abs/1312.6055`.

[26] Hilal Asi and John C Duchi. "The importance of better models in stochastic optimization". Proceedings of the National Academy of Sciences **116**, 22924–22930 (2019).

[27] Billy Jin, Katya Scheinberg, and Miaolan Xie. "High probability complexity bounds for line search based on stochastic oracles" (2021). arXiv:2106.06454.

[28] Jose Blanchet, Coralia Cartis, Matt Menickelly, and Katya Scheinberg. "Convergence rate analysis of a stochastic trust-region method via supermartingales". INFORMS Journal on Optimization **1**, 92–119 (2019).

[29] Courtney Paquette and Katya Scheinberg. "A stochastic line search method with expected complexity analysis". SIAM Journal on Optimization **30**, 349–376 (2020).

[30] Albert S Berahas, Liyuan Cao, and Katya Scheinberg. "Global convergence rate analysis of a generic line search algorithm with noise". SIAM Journal on Optimization **31**, 1489–1518 (2021).

[31] Coralia Cartis, Nicholas IM Gould, and Ph L Toint. "On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization problems". Siam journal on optimization **20**, 2833–2852 (2010).

[32] Coralia Cartis, Nicholas IM Gould, and Philippe L Toint. "On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization". SIAM Journal on Optimization **22**, 66–86 (2012).

[33] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. "Lower bounds for finding stationary points I". Mathematical Programming **184**, 71–120 (2020).

[34] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. ""convex until proven guilty": Dimension-free acceleration of gradient

descent on non-convex functions". In International Conference on Machine Learning. Pages 654–663. PMLR (2017).

[35] Chi Jin, Praneeth Netrapalli, and Michael I Jordan. "Accelerated gradient descent escapes saddle points faster than gradient descent". In Conference On Learning Theory. Pages 1042–1085. PMLR (2018). url: `https://proceedings.mlr.press/v75/jin18a.html`.

[36] Saeed Ghadimi and Guanghui Lan. "Stochastic first-and zeroth-order methods for nonconvex stochastic programming". SIAM Journal on Optimization **23**, 2341–2368 (2013).

[37] Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake Woodworth. "Lower bounds for non-convex stochastic optimization" (2019). arXiv:1912.02365.

[38] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator". In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems. Volume 31. Curran Associates, Inc. (2018). url: `https://proceedings.neurips.cc/paper/2018/file/1543843a4723ed2ab08e18053ae6dc5b-Paper.pdf`.

[39] Shiro Tamiya and Hayata Yamasaki. "Stochastic gradient line bayesian optimization: Reducing measurement shots in optimizing parameterized quantum circuits" (2021). arXiv:2111.07952.

[40] Pascual Jordan and Eugene Paul Wigner. "über das paulische äquivalenzverbot". In The Collected Works of Eugene Paul Wigner. Pages 109–129. Springer (1993).

[41] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. "Evaluating analytic gradients on quantum hardware". Physical Review A **99**, 032331 (2019).

[42] Joonho Lee, William J Huggins, Martin Head-Gordon, and K Birgitta Whaley. "Generalized unitary coupled cluster wave functions for quantum computation". Journal of Chemical Theory and Computation **15**, 311–324 (2018).

[43] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. "A variational eigenvalue solver on a photonic quantum processor". Nature Communications **5**, 1–7 (2014). url: `https://doi.org/10.1038/ncomms5213`.

[44] Ilya G Ryabinkin, Tzu-Ching Yen, Scott N Genin, and Artur F Izmaylov. "Qubit coupled cluster method: a systematic approach to quantum chemistry on a quantum computer". Journal of Chemical Theory and Computation **14**, 6317–6326 (2018).

[45] Ho Lun Tang, VO Shkolnikov, George S Barron, Harper R Grimsley, Nicholas J Mayhall, Edwin Barnes, and Sophia E Economou. "qubit-ADAPT-VQE: An adaptive algorithm for constructing hardware-efficient ansätze on a quantum processor". PRX Quantum **2**, 020310 (2021).

[46] Dmitry A. Fedorov, Yuri Alexeev, Stephen K. Gray, and Matthew Otten. "Unitary Selective Coupled-Cluster Method". Quantum **6**, 703 (2022).

[47] Pranav Gokhale, Olivia Angiuli, Yongshan Ding, Kaiwen Gui, Teague Tomesh, Martin Suchara, Margaret Martonosi, and Frederic T Chong. "$o(n^3)$ measurement cost for variational quantum eigensolver on molecular Hamiltonians". IEEE Transactions on Quantum Engineering **1**, 1–24 (2020).

[48] Ruobing Chen, Matt Menickelly, and Katya Scheinberg. "Stochastic optimization using a trust-region method and random models". Mathematical Programming **169**, 447–487 (2018).

[49] Léon Bottou, Frank E Curtis, and Jorge Nocedal. "Optimization methods for large-scale machine learning". Siam Review **60**, 223–311 (2018).

[50] Yoel Drori and Ohad Shamir. "The complexity of finding stationary points with stochastic gradient descent". In International Conference on Machine Learning. Pages 2658–2667. PMLR (2020). url: `https://proceedings.mlr.press/v119/drori20a.html`.

[51] Cong Fang, Zhouchen Lin, and Tong Zhang. "Sharp analysis for nonconvex SGD escaping from saddle points". In Conference on Learning Theory. Pages 1192–1234. PMLR (2019). url: `https://proceedings.mlr.press/v99/fang19a.html`.

[52] S Reddi, Manzil Zaheer, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. "Adaptive

methods for nonconvex optimization". In Proceedings of 32nd Conference on Neural Information Processing Systems (NIPS 2018). (2018). url: https://proceedings.neurips.cc/paper/2018/file/90365351ccc7437a1309dc64e4db32a3-Paper.pdf.

[53] Léon Bottou and Olivier Bousquet. "The tradeoffs of large scale learning". In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems. Volume 20. Curran Associates, Inc. (2007). url: https://proceedings.neurips.cc/paper/2007/file/0d3180d672e08b4c5312dcdafdf6ef36-Paper.pdf.

[54] Peter J Karalekas, Nikolas A Tezak, Eric C Peterson, Colm A Ryan, Marcus P da Silva, and Robert S Smith. "A quantum-classical cloud platform optimized for variational hybrid algorithms". Quantum Science and Technology **5**, 024003 (2020).

[55] H-J Briegel, Tommaso Calarco, Dieter Jaksch, Juan Ignacio Cirac, and Peter Zoller. "Quantum computing with neutral atoms". Journal of modern optics **47**, 415–451 (2000).

[56] Sergey Bravyi, Jay M Gambetta, Antonio Mezzacapo, and Kristan Temme. "Tapering off qubits to simulate fermionic Hamiltonians" (2017). arXiv:1701.08213.

[57] MD SAJID ANIS, Héctor Abraham, AduOffei, Rochisha Agarwal, Gabriele Agliardi, Merav Aharoni, Ismail Yunus Akhalwaya, Gadi Aleksandrowicz, Thomas Alexander, Matthew Amy, Sashwat Anagolum, Eli Arbel, Abraham Asfaw, Anish Athalye, Artur Avkhadiev, et al. "Qiskit: An open-source framework for quantum computing" (2021).

[58] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization". ACM Transactions on Mathematical Software (TOMS) **23**, 550–560 (1997).

[59] Raghu Bollapragada, Richard Byrd, and Jorge Nocedal. "Adaptive sampling strategies for stochastic optimization". SIAM Journal on Optimization **28**, 3312–3343 (2018).

[60] Raghu Bollapragada, Jorge Nocedal, Dheevatsa Mudigere, Hao-Jun Shi, and Ping Tak Peter Tang. "A progressive batching L-BFGS method for machine learning". In International Conference on Machine Learning. Pages 620–629. PMLR (2018). url: https://proceedings.mlr.press/v80/bollapragada18a.html.

[61] Raghu Pasupathy, Peter Glynn, Soumyadip Ghosh, and Fatemeh S Hashemi. "On sampling rates in simulation-based recursions". SIAM Journal on Optimization **28**, 45–73 (2018).

[62] Andrew Arrasmith, Lukasz Cincio, Rolando D Somma, and Patrick J Coles. "Operator sampling for shot-frugal optimization in variational algorithms" (2020). arXiv:2004.06252.

[63] Yangyang Xu and Wotao Yin. "Block stochastic gradient iteration for convex and nonconvex optimization". SIAM Journal on Optimization **25**, 1686–1716 (2015).

A Appendix

A.1 Derivation of (7)

We require $g^k(\theta^k;\xi^k)$ to satisfy the following property.

Condition A.1. *For some $p' \in (0,1/2)$ and for some accuracy level $\epsilon_g > 0$,*

$$\mathbb{P}_{\xi^k}\left[e_g(\theta^k;\xi^k) \leq \max\{\epsilon_g, L_g\alpha_k\|g(\theta^k;\xi^k)\|\}\right] \geq 1-p', \tag{17}$$

where $e_g(\theta^k;\xi^k)$ denotes $\|g(\theta^k;\xi^k) - \nabla f(\theta^k)\|$ and L_g denotes a local Lipschitz constant of the gradient $\nabla f(\theta^k)$.

In other words, Condition A.1 requires that, with sufficiently high probability, the error in the estimator be bounded by the Lipschitz constant times $\|\mathbf{s}^k - \theta^k\|$, an upper bound on the change in the gradient size.

Applying Chebyshev's inequality, we can guarantee a gradient estimate g^k satisfying Condition A.1 provided we choose each $N_{g_i,k}$ according to (7). Indeed, (7) is the number of independent observations needed to attain

$$\mathbb{P}_{\xi^k}\left[e_{g_i}(\theta^k;\xi^k) \leq \max\{\epsilon_g, L_i\alpha_k|g_i(\theta^k;\xi^k)|\}\right] \geq 1-p,$$

where $e_{g_i}(\theta^k;\xi^k) = |g_i(\theta^k;\xi^k) - \partial_i f(\theta^k)|$. Then, by a union bound, with probability $1-p^n \geq 1-p'$,

$$e_g(\theta^k;\xi^k) \leq \max\{\epsilon_g, L_g\alpha_k\|g(\theta^k;\xi^k)\|\}.$$

A.2 Deriving $N_{f,k}$

We now state our condition on $f(\theta^k;\xi^k)$. The condition for $f(\mathbf{s}^k;\xi^k)$ is analogous.

Condition A.2. *Define the* estimation error *as* $e(\theta^k;\xi^k) := |f(\theta^k) - f(\theta^k;\xi^k)|$.

The estimation error is a one-sided subexponential-like random variable with mean bounded by a constant $\epsilon_f > 0$. That is, there exist parameters (ν,β) such that

$$\mathbb{E}_{\xi^k}\left[e(\theta^k;\xi^k)\right] \le \epsilon_f \tag{18}$$

and

$$\mathbb{E}_{\xi^k}\left[\exp\left\{\lambda\left(e(\theta^k;\xi^k) - \mathbb{E}_{\xi^k}\left[e(\theta^k;\xi^k)\right]\right)\right\}\right] \le \exp\left(\frac{\lambda^2\nu^2}{2}\right), \quad \forall\lambda\in\left[0,\frac{1}{\beta}\right]. \tag{19}$$

When (18) *and* (19) *are satisfied, we say that* $e(\theta^k;\xi^k)$ is (ν,β)-subexponential.

The following corollary demonstrates that, for any $\epsilon_f > 0$, a sufficiently large sample size exists such that Condition A.2 holds.

Corollary A.1. *For any given $\epsilon_f > 0$ we may choose a sample size $N_{f,k}$ such that*

$$\epsilon_f = \sqrt{\frac{V}{N_{f,k}}}$$

(where V is as in Proposition A.1) in order to guarantee $e(\theta^k;\xi^k)$ is (ν,β)-subexponential with

$$\nu = \beta = 8\exp(2)\max\left\{\sqrt{\frac{2\mathbb{V}_\xi[e(\theta^k;\xi)]}{N_{f,k}}}, 2(C+\epsilon_f)\right\}, \tag{20}$$

and

$$\mathbb{E}_{\xi^k}[e(\theta^k;\xi^k)] \le \epsilon_f,$$

where C is as in Proposition A.2.

We now prove Corollary A.1. We first record a result that was proven in [27, Proposition 1].

Proposition A.1. *Suppose $e(\theta^k;\xi)$ is a $(\hat\nu,\hat\beta)$-subexponential random variable and $\mathbb{V}_\xi[f(\theta^k;\xi)] \le V$. Then*

$$\mathbb{E}_{\xi^k}[e(\theta^k;\xi^k)] \le \sqrt{\frac{V}{N_{f,k}}}$$

and $e(\theta^k;\xi^k)$ is (ν,β)-subexponential, with $\nu = \beta = 8\exp(2)\max\left\{\frac{\hat\nu}{\sqrt{N_{f,k}}},\hat\beta\right\}$.

Because of the form of the single-shot estimators $f(\theta^k;\xi)$ in the parameter shift setting (a finite weighted sum of Bernoulli random variables), we have that $|e(\theta^k;\xi)| \le C$ deterministically, where $C = 2\sum_i |a_i|$. Thus, we can show the following result.

Proposition A.2. *$e(\theta^k;\xi)$ is a $(\sqrt{2\mathbb{V}\left[e(\theta^k;\xi)\right]}, 2(C+\epsilon_f))$-subexponential random variable.*

Proof. Let $Y = \dfrac{e(\theta^k;\xi) - \mathbb{E}\left[e(\theta^k;\xi)\right]}{C+\epsilon_f}$, and suppose (as in Condition A.2) that $\mathbb{E}\left[e(\theta^k;\xi)\right] \le \epsilon_f$. By definition of variance, $\mathbb{E}\left[Y^2\right] = \frac{1}{(C+\epsilon_f)^2}\mathbb{V}\left[e(\theta^k;\xi)\right] := V$. Moreover,

$$|Y| \le \frac{|e(\theta^k;\xi)| + \mathbb{E}\left[e(\theta^k;\xi)\right]}{C+\epsilon_f} \le 1.$$

Thus, bounding the moment generating function,

$$\mathbb{E}\left[\exp(\lambda Y)\right] = \mathbb{E}\left[\sum_{k=0}^{\infty}\frac{\lambda^k Y^k}{k!}\right] \le 1 + \sum_{k=2}^{\infty}\frac{\lambda^k}{k!}V \le 1 + \left(\sum_{k=0}^{\infty}\lambda^k\right)\frac{\lambda^2 V}{2}.$$

For all $\lambda < 1$, the geometric series on the right will converge; moreover, for $\lambda \le \frac{1}{2}$,

$$\mathbb{E}\left[\exp(\lambda Y)\right] \le \exp\left(\frac{\lambda^2(\sqrt{2V})^2}{2}\right).$$

That is, $\frac{e(\theta^k;\xi)}{C+\epsilon_f}$ is a $(\sqrt{2V},2)$-subexponential random variable. The result follows. ☐

Because the variance $\mathbb{V}_\xi\left[f(\theta^k;\xi)\right]$ is bounded in the parameter-shift setting, $(f(\theta^k;\xi)$ can be expressed as a finite weighted sum of Bernoulli random variables), Proposition A.1 and Proposition A.2 together give us Corollary A.1.

H_2	Shots				Switches				Communications			
Solver	Q_{25}	Q_{50}	Q_{75}	mean	Q_{25}	Q_{50}	Q_{75}	mean	Q_{25}	Q_{50}	Q_{75}	mean
SHOALS	4.61×10^5	2.31×10^6	1.42×10^7	9.09×10^6	400	**472**	**815**	**662**	23	28	**45**	**36**
iCANS	$\mathbf{2.74 \times 10^5}$	6.83×10^5	1.44×10^6	1.55×10^6	1200	1820	2480	1965	30	46	62	49
Adam-100	5.68×10^5	7.34×10^5	$\mathbf{9.44 \times 10^5}$	$\mathbf{7.36 \times 10^5}$	5720	7380	9480	7403	143	185	237	185
Adam-1000	5.64×10^6	7.12×10^6	9.00×10^6	7.13×10^6	5680	7160	9040	7173	142	179	226	179
LBFGS-100	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
LBFGS-1000	3.15×10^5	$\mathbf{6.53 \times 10^5}$	∞	∞	**315**	652	∞	∞	**7**	**15**	∞	∞
LiH												
SHOALS	7.92×10^8	1.27×10^9	1.69×10^9	1.41×10^9	$\mathbf{2.41 \times 10^5}$	$\mathbf{3.47 \times 10^5}$	$\mathbf{4.66 \times 10^5}$	$\mathbf{3.72 \times 10^5}$	**79**	**110**	**149**	**122**
iCANS	2.06×10^9	2.75×10^9	4.16×10^9	2.86×10^9	1.82×10^7	2.04×10^7	2.59×10^6	2.15×10^6	219	245	311	259
Adam-100	$\mathbf{1.61 \times 10^8}$	$\mathbf{2.25 \times 10^8}$	$\mathbf{3.29 \times 10^8}$	$\mathbf{2.36 \times 10^8}$	1.61×10^6	2.25×10^6	3.29×10^6	2.36×10^6	193	271	396	283
Adam-1000	1.11×10^9	1.77×10^9	2.38×10^9	1.83×10^9	1.11×10^6	1.77×10^6	2.38×10^6	1.83×10^6	134	213	286	219
H_4												
SHOALS	1.24×10^{11}	1.36×10^{11}	1.46×10^{11}	1.35×10^{11}	$\mathbf{1.51 \times 10^7}$	$\mathbf{1.66 \times 10^7}$	$\mathbf{1.71 \times 10^7}$	$\mathbf{1.61 \times 10^7}$	**482**	**525**	**531**	**508**
iCANS	9.39×10^{11}	1.04×10^{12}	1.14×10^{12}	1.04×10^{12}	3.27×10^8	3.43×10^8	3.59×10^8	3.43×10^8	3086	3237	3388	3237
Adam-100	$\mathbf{1.29 \times 10^{10}}$	$\mathbf{1.39 \times 10^{10}}$	$\mathbf{1.46 \times 10^{10}}$	$\mathbf{1.38 \times 10^{10}}$	1.29×10^8	1.39×10^8	1.46×10^8	1.38×10^8	1229	1309	1380	1301
Adam-1000	1.09×10^{11}	1.32×10^{11}	1.75×10^{11}	1.41×10^{11}	1.09×10^8	1.32×10^8	1.75×10^8	1.41×10^8	1032	1240	1649	1329

Table 3: For each tested molecule and for each solver, we display the 25^{th}, 50^{th}, and 75^{th} percentiles (Q_{25}, Q_{50}, Q_{75}, respectively), as well as the mean, of the numbers of shots, switches, and communications needed to attain chemical accuracy in our experiments. Lowest values are highlighted in boldface font. We note that LBFGS-100 and LBFGS-1000 have undefined or "infinite" values for each summary statistic for the two larger molecules and are hence excluded entirely.