

Adaptively measuring quantum expectation  
values using the empirical Bernstein  
stopping rule

Bachelorarbeit

am Institut für theoretische Physik, Düsseldorf  
vorgelegt von

Ugur Tepe

Matrikelnummer: 2724342

**Erstprüfer**

Prof. Dr. Martin Kliesch  
Institute for Quantum Inspired and  
Quantum Optimisation  
Hamburg University of Technology

**Zweitprüfer:**

Prof. Dr. Dagmar Bruß  
Institut für theoretische Physik III  
Heinrich-Heine-Universität  
Düsseldorf

27.11.2023

## Abstract

Quantum computing promises to be exponentially faster than classical computers, in a wide array of computational tasks. However, the current generation of quantum hardware is not capable of achieving such feats, due to noise limitations.

Methods such as variational quantum algorithms (VQA) are developed with the goal of extracting the most out of currently available quantum hardware. While VQAs are a popular and promising approach, they face a multitude of issues that need to be solved. The issue this thesis is concerned with is the reduction of the measurement effort when measuring an expectation value up to certain accuracy.

Hence, the goal of this thesis is to utilise an adaptive stopping algorithm, the so-called empirical Bernstein stopping (EBS) algorithm, to reduce the measurement effort. As EBS makes use of empirical variance information, it promises better results than more commonly used non-adaptive alternatives, such as Hoeffding's inequality or a simple sample budget.

Reducing the measurement effort is, however, not a unique problem of VQAs and therefore the results of this thesis will be applicable to a wider array of quantum algorithms.

# Contents

<b>1. Introduction</b>	<b>1</b>
<b>2. Preliminaries</b>	<b>3</b>
2.1. Probability Theory . . . . .	3
2.1.1. Random Variables . . . . .	3
2.1.2. Expectation Value . . . . .	4
2.1.3. Variance . . . . .	4
2.2. Concentration Inequalities . . . . .	5
2.2.1. Markov's Inequality . . . . .	5
2.2.2. Höffding's Inequality . . . . .	5
2.2.3. Bernstein's Inequality . . . . .	6
2.2.4. Höffding's vs. Bernstein's Inequality . . . . .	7
2.3. Quantum mechanics . . . . .	8
2.3.1. Schrödinger's Equation . . . . .	8
2.3.2. Operators . . . . .	9
2.3.3. Operator Exponential . . . . .	10
2.4. Quantum computing . . . . .	10
2.4.1. Qubits . . . . .	11
2.4.2. Pauli Operators/ Matrices . . . . .	13
2.4.3. Quantum Gates . . . . .	13
2.4.4. Measurements in Different Bases . . . . .	17
2.4.5. Variational Quantum Algorithms . . . . .	17
<b>3. Methods</b>	<b>22</b>
3.1. Empirical Bernstein Stopping (EBS) . . . . .	22
3.1.1. $(\epsilon, \delta)$ -Stopping Rules . . . . .	22
3.1.2. Empirical Bernstein Stopping Rule . . . . .	23
3.1.3. Base Algorithm . . . . .	24
3.1.4. Modifications on EBA . . . . .	26
3.1.5. Effect of the Variance on EBS . . . . .	28
3.2. Constructing a suitable VQE ansatz . . . . .	30
3.2.1. Quantum Decomposition . . . . .	30
3.2.2. Hamiltonian to Parametrized Circuit . . . . .	31
3.2.3. EBA inside a VQE . . . . .	32
<b>4. Results</b>	<b>33</b>
4.1. Toy Example: Two Qubit System . . . . .	33
4.1.1. Effect of $\epsilon$ on Actual Accuracy and Success Probability . . . . .	36
4.1.2. Effect of $\epsilon$ on Sample Complexity . . . . .	37
4.2. Dissociation Curve of $H_2$ . . . . .	40
4.2.1. VQE run for fixed $d$ . . . . .	41
4.2.2. Comparing Measurement Strategies . . . . .	42
4.2.3. Effect of $\epsilon$ on Actual Accuracy and Success Probability . . . . .	43
4.2.4. Effect of $\epsilon$ on Sample Complexity . . . . .	44
4.2.5. Dissociation Curve . . . . .	46
<b>5. Conclusion</b>	<b>48</b>
<b>6. Outlook</b>	<b>49</b>

<b>A. Bernstein Bound Derivation</b>	<b>50</b>
<b>B. Quantum Mechanics/ Computing</b>	<b>51</b>
B.1. Exponential Operator Derivation . . . . .	51
B.2. Parameter Shift Rule . . . . .	51
B.3. Quantum Gates . . . . .	52
<b>C. Quantum Circuit Decomposition: Example Circuit</b>	<b>53</b>
<b>D. Coefficients <math>\{g_i\}</math> for eq. (105)</b>	<b>53</b>

# 1. Introduction

Although the current generation of quantum computers can exceed classical computers in tasks specifically designed for available quantum hardware [1, 2], a practical quantum advantage is yet to be achieved. Thus, the goal is to utilise the available hardware, often referred to as noisy intermediate scale quantum era (NISQ) hardware, to achieve said advantage over classical computers. These NISQ quantum computers are limited in the number of qubits and are subject to noise. The latter effectively restricts the total length of the computation as well as the total number of operations used.

Fault-tolerant (FT) quantum computers are the solution for the noise problem, however, they require more qubits than currently available. Additionally, adding more qubits and increasing circuit depth to implement FT quantum computers also increases noise.

Variational quantum algorithms (VQAs) are a popular approach to employ the current generations of quantum hardware. These algorithms are designed to obey the limited circuit depth and qubit numbers. The key idea behind VQAs is to utilise classical computers that interface with the quantum computer via the parameters of a parameterised quantum circuit. In particular, these parameters are optimised using classical optimisation techniques, e.g., a gradient-based one. These parameters are tuned such that the cost function is minimised. The cost function can be, for example, the expected energy of the Hamiltonian as a function of some parameter(s).

This is possible due to the variational principle, which states that any measured energy is always above the ground energy. Thus, optimising towards the ground energy is effective in finding the ground state or any other minima.

Naturally, such methods can be used to solve problems from quantum chemistry or problems of combinatorial nature [3]. However, these VQAs face several issues:

- The optimisation process can get stuck in a barren plateau or local minima [4, 5],
- The sample effort for the read-out of the algorithm can bottleneck the VQA [3, 6],
- In general, finding a suitable parametrized circuit (often referred to as the ansatz) is hard. [7].

Barren plateaus are regions in the cost landscape where the expected gradient goes to zero with increased system size, causing the optimiser to get stuck. A similar problem can be encountered in local minima as the gradient vanishes there as well.

For the barren plateau problem there are multiple suggested techniques to avoid them [8–10].

Especially in quantum chemistry problems, a certain accuracy is required which results in a sample bottleneck <sup>1</sup> as naturally one wants to also minimise the amounts of samples that are required for an accurate estimate. Such estimates are produced using so-called stopping algorithms, which guarantee an accuracy and confidence. They determine when to stop sampling according

---

<sup>1</sup>Measuring a state leads to its collapse, hence it has to be re-prepared for additional measurements.

to a specified condition, e.g., the accuracy of the current estimate.

One such algorithm is the empirical Bernstein stopping (EBS) algorithm. Stopping rules, like EBS, manage how many samples are needed to estimate the mean of a series of i.i.d random variables. This estimate is called an  $(\epsilon, \delta)$ -estimate, as it is within  $\epsilon$  of the actual mean, with the probability of at least  $1 - \delta$ . In contrast to similar stopping rules, EBS incorporates the empirical variance, which is estimated as well, into its stopping condition. Hence, EBS is called an adaptive stopping rule, as it is dependent on the explicit samples drawn.

The goal of this thesis is to test whether EBS decrease the sample requirements for quantum algorithms such as VQAs. EBS performs well, especially when the variance of the random variable is low compared to their range. VQAs provide a wide array of variance regimes, as near a minima or saddle point the variance vanishes while increasing in other regions.

As the overarching goal of VQAs is to estimate the ground energy, EBS may provide a significant improvement compared to usual methods, as it benefits from the vanishing variance. Over a VQE optimisation run, this may lead to a more efficient sampling strategy.

## 2. Preliminaries

This section serves as a quick primer on topics such as probability theory, quantum mechanics and quantum computing. Firstly, a refresher on probability theory is given, as concepts like random variables and expected values are a key concepts in this thesis. Further, probability theory is needed to introduce the concept of concentration inequalities, on which the empirical Bernstein stopping algorithm is based on.

Secondly, quantum mechanics serves as a foundation to quantum computing, where concepts as qubits, gates and circuits are introduced as well as variational quantum algorithms.

### 2.1. Probability Theory

This section provides a refresher on the basics of probability theory. Probability theory is a fundamental component of quantum mechanics, quantum information. Alongside the basics of probability theory, this section also introduces the concept of tail bounds, starting with the most fundamental one. Tail bounds are an important conceptual tool to understand as they are employed by stopping algorithms which are introduced and discussed in latter sections of this thesis.

#### 2.1.1. Random Variables

A random variable  $X$  is a function that maps from an (unknown) probability space  $S$  to a real number  $\mathbb{R}$  [11].

$$X : S \rightarrow \mathbb{R} \quad (1)$$

Random variables can be continuous or discrete, however in this thesis only discrete random variables are of importance. The randomness of the random variable is tied to the random experiment they come from. For example, in the case of a die, the trivial random variable looks like:

$$x(1) = 1, \quad x(2) = 2, \quad x(3) = 3, \quad x(4) = 4, \quad x(5) = 5, \quad x(6) = 6 \quad (2)$$

with  $X = \{x(i)\}$  and  $i = \{1, 2, \dots, 6\}$ .

Each random variable  $x_i$  occurs with a probability of  $1/6$  and has the value of the corresponding die number.

Another example would be to divide the outcomes of the dice into odd and even facing numbers. This would result in the corresponding random variable only having 2 values, **odd** or **even**:

$$x(\text{odd}) = -1 \quad x(\text{even}) = 1 \quad (3)$$

with  $X = \{x(\text{odd}), x(\text{even})\}$ .

Each of these two random variables has the associated probability of  $1/2$  and discrete number, 1 or  $-1$ , mapped to them.

When a collection of identical random variables is considered, and it is assumed that each one is independent of the other, they are called independent and identically distributed random variables (i.i.d.).

Multiple dice throws, for example, are i.i.d. random variables.

### 2.1.2. Expectation Value

The expected value of a random variable is the average of every outcome that can occur multiplied by its probability density. It is also known as the first moment (because  $X$  only goes in linearly).

In the discrete case, it is defined as:

$$\mathbb{E}[X] = \langle X \rangle = \mu = \sum_i x_i p_i, \quad (4)$$

with  $p_i$  being the probability of  $x_i$ .

Of course calculating the expected value like this is not always possible as the probability distribution is usually unknown, to that extent one can use the so-called empirical mean. The aforementioned is calculated as follows:

$$\bar{X} = \frac{1}{N} \sum_{i=1}^N X_i. \quad (5)$$

### 2.1.3. Variance

The variance measures how much the random variable may vary from its expected value. The variance of a random variable  $X$  is the expected squared deviation from its expected value  $\mathbb{E}[X] = \mu$ :

$$\text{Var}(X) = \mathbb{E}[(X - \mu)^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2. \quad (6)$$

For a discrete random variable  $X$ , assuming the probabilities are known, its variance can be calculated as:

$$\text{Var}(X) = \sum_i p_i (x_i - \mu)^2. \quad (7)$$

In many applications it is not possible to access the probabilities  $p_i$  to calculate the variance. For these cases, there is the so-called (unbiased) **sample variance**  $\text{Var}_s$ . Like the name suggests, the sample variance is calculated from  $N$  samples:

$$\text{Var}_s(X) = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{X})^2, \quad (8)$$

with  $\bar{X}$  being the empirical mean.

Closely related to the variance is the **standard deviation**,  $\sigma$ :

$$\sigma = \sqrt{\text{Var}(X)}. \quad (9)$$

Hence, the variance is also commonly denoted as  $\sigma^2$ .

The variance of a bounded probability distribution can be bounded using Popoviciu's inequality [12]:

$$0 \leq \sigma^2 \leq \frac{R^2}{4}. \quad (10)$$



## 2.2. Concentration Inequalities

When considering a series of random variables, it may be of use to determine how far they deviate from their expected value based on the number of samples. It is clear that the empirical mean of this sequence converges to the actual mean or expected value, based on how many samples are drawn. Often it is of interest to know when the empirical mean is converged such that it is within a constant from the actual mean. This convergence can be characterised using so-called **concentration inequalities**.

These inequalities vary in how tightly they bound the tail and the restrictions imposed on the random variables. The most fundamental inequality is Markov's, it assumes only that the random variables are almost surely non-negative<sup>2</sup>. Markov's inequality serves as an introduction to more advanced inequalities like Bernstein's inequality and Höfdding's inequality are also introduced.

### 2.2.1. Markov's Inequality

Let  $X$  be an almost surely non-negative random variable and  $\epsilon > 0$ , then Markov's inequality is as follows:

$$\mathbb{P}[X \geq \epsilon] \leq \frac{\mathbb{E}[X]}{\epsilon}. \quad (11)$$

With this inequality the upper bound to probabilities of various events can be calculated, e.g. how likely it is at most to roll a die that is greater or equal to a 4.

$$\mathbb{P}[X \geq 4] \leq \frac{3.5}{4} = 0.875. \quad (12)$$

The 87.5% is only a crude upper bound for the probability. Compared to the actual probability  $\mathbb{P}[X \geq 4] = 50\%$ , the upper bound provided by Markov's inequality is substantially larger. This discrepancy is due to the fact that Markov's inequality assumes only that the random variables are almost surely non-negative. If more assumptions are made, or even more empirical information is taken into account, these bounds can be improved significantly. The subject of the following subsection are such methods.

### 2.2.2. Höfdding's Inequality

An inequality frequently used for stopping algorithms, e.g., in machine learning or quantum computing, is Höfdding's inequality [13]. Höfdding's inequality gives a bound on how much the mean of a series of random variables deviates from its expected value  $\mu$ . This bound depends on their range  $R$  and the number of samples  $t$  taken.

Let  $X_1, \dots, X_t$  be independent random variables with  $a_i \leq X_i \leq b_i$  and  $\bar{X}_t = \frac{1}{t} \sum_{i=1}^t X_i$ , then Höfdding's inequality [14] is as follows:

$$\mathbb{P}[|\bar{X}_t - \mu| \geq \epsilon] \leq 2 \exp \left( -\frac{2(t\epsilon)^2}{\sum_{i=1}^t (b_i - a_i)^2} \right). \quad (13)$$

---

<sup>2</sup>Almost surely non-negative random variables are random variables that fulfil:  $\mathbb{P}[X \geq 0] = 1$ . This means they can take negative values, but with the probability of 0 to occur.

If all  $X_i$  are bounded by the same interval  $[a_i, b_i]$  with the range  $R = b - a$ , equation (13) simplifies to:

$$\mathbb{P}[|\bar{X}_t - \mu| \geq \epsilon] \leq 2 \exp\left(-\frac{2t\epsilon^2}{R^2}\right) =: \delta, \quad (14)$$

with  $\sum_{i=1}^t (b_i - a_i)^2 = tR^2$  and defining the left-hand side as  $\delta$ .

$\delta$  is the probability that after  $t$  many samples, the empirical mean deviates at-least  $\epsilon$  from the actual mean. This inequality can be rearranged to a bound form [15], which is more suitable to quantify the current accuracy of the estimate:

$$|\bar{X}_t - \mu| \leq R \sqrt{\frac{\ln(2/\delta)}{2t}}. \quad (15)$$

This inequality holds with probability of at-least  $1 - \delta$ . Since Höfdding's inequality depends only on the range  $R$ , we can set the right-hand side of eq. (15) equal to a desired accuracy  $\epsilon$  and confidence  $1 - \delta$  and solve for  $t$ .

$$t_{\min} := 2 \left(\frac{R}{\epsilon}\right)^2 \ln(2/\delta). \quad (16)$$

$t_{\min}$  is also referred to as the sample complexity of the random variable, as it determines how many samples are needed to obtain an estimate that is within  $\epsilon$ ,  $1 - \delta$  of the time. This concept of having an estimate, characterized by the two parameters  $\epsilon$  and  $\delta$  is called an  $(\epsilon, \delta)$  estimate and will be discussed in a latter section (see sec. 3.1.1) in more detail.

### 2.2.3. Bernstein's Inequality

Similar to Höfdding's inequality, Bernstein's inequality also bounds the sum of random variables. The key difference is that unlike Höfdding's inequality, Bernstein's is dependent on the variance of the random variables as well. Let  $X_1, \dots, X_t$  be i.i.d. random variables with range  $R$ , expected value  $\mu$ , and variance  $\sigma^2$  each.

Let  $\bar{X}_t = \frac{1}{t} \sum_{i=1}^t X_i$  and  $\Sigma^2 = \sum_{i=1}^t \sigma_i^2$ . Then, Bernstein's inequality [14] is as follows:

$$\mathbb{P}[|\bar{X}_t - \mu| \geq \epsilon] \leq 2 \exp\left(-\frac{(t\epsilon)^2/2}{\Sigma^2 + Rt\epsilon/3}\right) \stackrel{!}{=} \delta. \quad (17)$$

Like with Höfdding's inequality, it is beneficial to rewrite the inequality into a bounded form. The derivation of this bound can be found in appendix A and yields:

$$|\bar{X}_t - \mu| \leq \underbrace{\sigma \sqrt{\frac{2 \ln(2/\delta)}{t}}}_{\mathcal{O}(1/\sqrt{t})} + \underbrace{\frac{R \ln(2/\delta)}{3t}}_{\mathcal{O}(1/t)}. \quad (18)$$

Because the variance is unknown in practical scenarios (such as the underlying distribution), the variance in equation (18) has to be replaced [16] by the empirical standard deviation  $\bar{\sigma}_t = \sqrt{\frac{1}{t} \sum_{i=1}^t (X_i - \bar{X}_t)^2}$ . This yields the empirical form of eq. (18):

$$|\bar{X}_t - \mu| \leq \bar{\sigma}_t \sqrt{\frac{2 \ln(3/\delta)}{t}} + \frac{3R \ln(3/\delta)}{t}. \quad (19)$$

A proof of the inequality and bound can be found in [16]. The fact that the variance can be replaced with an estimated variance in Bernstein's bound is a surprising and profound statement. This result makes it possible to use Bernstein's bound in cases where the probability distribution and the corresponding variance is not known.

In the following section, the previously introduced bounds of Bernstein and Höfdding are quickly compared with each other.

#### 2.2.4. Höfdding's vs. Bernstein's Inequality

To better understand in which scenarios Bernstein's bound is advantageous over Höfdding's bound, it is useful to compare the inequalities. In the empirical Bernstein bound (18), the term with the empirical standard deviation or the empirical variance decreases with  $1/\sqrt{i}$  while the term with the range  $R$  decreases with  $1/t$ , rendering the term that involves the range negligible for large  $t$ . If the standard deviation is substantially smaller than the range, Bernstein's bound becomes much tighter than Höfdding's bound for larger  $t$ . In this case, the remaining term both decrease with  $\mathcal{O}(1/\sqrt{i})$ . Hence, the advantage of one bound over the other depends on whether  $\sigma \ll R$  is fulfilled.

$$|\bar{X}_t - \mu| \leq \sigma \sqrt{\frac{2 \ln(3/\delta)}{t}} + \frac{R \ln(2/\delta)}{3t} \quad (\text{Bernstein's Bound}) \quad (20)$$

$$|\bar{X}_t - \mu| \leq R \sqrt{\frac{\ln(2/\delta)}{2t}} \quad (\text{Höfdding's Bound}) \quad (21)$$

with  $\delta$  fixed to 0.1.

We plot the comparison of the two bounds (20) & (21) as a function of the sample number  $t$ , the range  $R$  and the variance  $\sigma^2$  in figure 1.

Note that the variance of any bounded probability distribution is bounded by  $R^2/4$ , see eq. (10). Hence, for a range of  $R = 2$  the maximally possible variance is  $\sigma^2 = 1$ , which is the worst-case for Bernstein's bound. Consequently, Höfdding's bound is seen to be tighter than Bernstein's bound in said worst-case, in figure 1. However, Bernstein's bound remained tighter for larger  $t$ , for the remaining cases. Additionally, we observe that Höfdding's bound does not depend on  $\sigma^2$ , hence does not benefit from decreasing variance as Bernstein's bound does. In the case where the variance is set constant and the range  $R$  is increased, apart from the worst-case scenario, Bernstein's bound becomes tighter as the condition  $\sigma \ll R$  is approached.

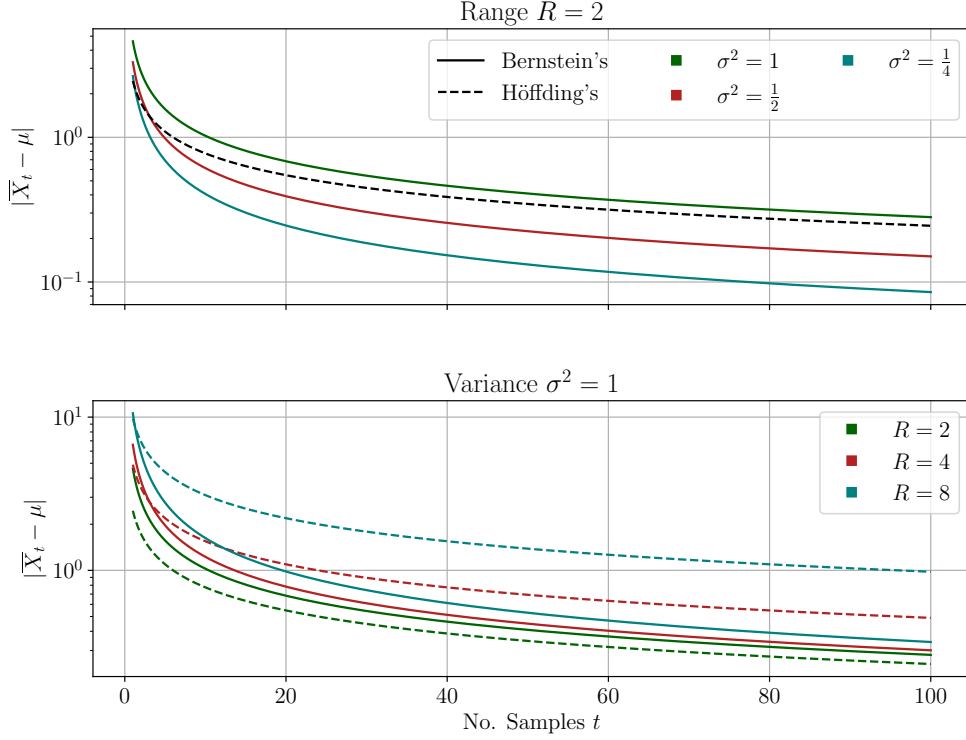


Figure 1: Bernstein's (20) and Hoeffding's bound (21) as a function of the number of samples  $t$  and the range  $R$ . Different values of  $R$  and  $\sigma$  are indicated by the corresponding colours respectively. Bernstein's bound is represented by a solid line, Hoeffding's with a dashed line. Further, the range is set to  $R = 2$  in the top plot, while the variance is set to  $\sigma^2 = R^2/4 = 1$  in the bottom figure. Note that in case of  $R = 2$  and  $\sigma^2 = 1$ , Hoeffding's bound is tighter than Bernstein's, while for the other scenarios Bernstein's bound remain tighter for large  $t$ . The right-hand sides of the corresponding equations are plotted, see equations (20) & (21).

### 2.3. Quantum mechanics

This section gives a brief recap of quantum mechanics such as operators, Pauli matrices and two-level systems, while also briefly introducing the tensor product. With these concepts, quantum circuits and quantum gates are introduced. As a follow-up, variational quantum algorithms are introduced, including the gradient-descent method to classically update the circuit's parameters.

#### 2.3.1. Schrödinger's Equation

Quantum states can be represented using a state vector  $|\psi\rangle$ . This state vector contains all information of that particular system. Similar to Newton's second law, one can use Schrödinger's equation to calculate the time evolution of a given quantum state  $|\psi\rangle$  based on a Hamiltonian  $\hat{H}$ . Schrödinger's equation is the fundamental equation in quantum mechanics and reads as follows:

$$i\hbar \frac{\partial |\psi(t)\rangle}{\partial t} = \hat{H} |\psi(t)\rangle. \quad (22)$$

The equation above is the time **time dependent** Schrödinger's equation. The **time independent** equation is as follows:

$$\hat{H}|\psi\rangle = E|\psi\rangle,$$

with  $E$  being the energy of the state  $|\psi\rangle$ .

Solving the time in dependent Schrödinger's equation yields a state that is independent of time. To obtain a time evolution of such state, the unitary **time evolution operator**  $\hat{U}(t - t_0)$  may be used. This operator propagates a state at a given time  $t_0$  to a latter time  $t$ .

$$\begin{aligned}\hat{U}(t, t_0) &= e^{-\frac{i\hat{H}}{\hbar}(t-t_0)} \\ |\psi(t)\rangle &= \hat{U}(t, t_0) |\psi(t_0)\rangle \\ &= e^{-\frac{i\hat{H}}{\hbar}(t-t_0)} |\psi(t_0)\rangle\end{aligned}\tag{23}$$

The state vector itself does not induce a probability density, only the modulus squared has physical meaning, as it is interpreted as a probability  $p_i$ :

$$p_i = |\langle i|\psi\rangle|^2.\tag{24}$$

### 2.3.2. Operators

An operator  $\hat{A}$  is a linear function that maps a state  $|\psi\rangle$ , onto another state  $|\phi\rangle$ .

$$\hat{A}|\psi\rangle = |\phi\rangle,\tag{25}$$

if  $|\psi\rangle \propto |\phi\rangle$  then  $|\psi\rangle$  is an eigenstate of operator  $\hat{A}$ , i.e.:

$$\hat{A}|\psi\rangle = \lambda|\psi\rangle,\tag{26}$$

where  $\lambda$  the corresponding eigenvalue.

Two distinct eigenvectors of an operator are orthogonal to each other, if the operator is normalised. Concisely, with the Kronecker  $\delta$ , this means that:

$$\langle\psi_i|\psi_j\rangle = \delta_{ij}.\tag{27}$$

In quantum mechanics, only eigenvalues are possible values for the outcomes of the corresponding observable  $\hat{A}$ . Any such operator that corresponds to an observable has to be linear and Hermitian, to ensures real eigenvalues.

$$\hat{A}(|f\rangle + |g\rangle) = (\hat{A}|f\rangle) + (\hat{A}|g\rangle)\tag{28}$$

$$\hat{A} = \hat{A}^\dagger\tag{29}$$

Being linear, operators can also be expressed as matrices <sup>3</sup>:

$$\hat{A} = \begin{bmatrix} \langle\psi_1|\hat{A}|\psi_1\rangle & \dots & \langle\psi_1|\hat{A}|\psi_m\rangle \\ \langle\psi_2|\hat{A}|\psi_1\rangle & \dots & \langle\psi_2|\hat{A}|\psi_m\rangle \\ \vdots & \vdots & \vdots \\ \langle\psi_m|\hat{A}|\psi_1\rangle & \dots & \langle\psi_m|\hat{A}|\psi_m\rangle \end{bmatrix} = \begin{bmatrix} A_{11} & \dots & A_{1m} \\ A_{21} & \dots & A_{2m} \\ \vdots & \vdots & \vdots \\ A_{m1} & \dots & A_{mm} \end{bmatrix}\tag{30}$$

---

<sup>3</sup>This is only possible if the corresponding Hilbert-space is finite dimensional, e.g., when considering a finite number of basis states.

with  $m < \infty$  and  $\psi_i, \psi_j$  being basis states.

Expressing operators as matrices will come in handy in latter sections (see. sec. 2.4.3). The choice of the basis states is arbitrary. For instance, if  $|\psi_i\rangle$  and  $|\psi_j\rangle$  are chosen to be eigenstates of the operator,  $A$  the corresponding matrix is diagonal.

The expectation value of the operator  $\hat{A}$  is given by:

$$\langle \hat{A} \rangle_\varphi = \langle \varphi | \hat{A} | \varphi \rangle. \quad (31)$$

Because expected values are always in relation to a particular state  $\varphi$ , we denote the state as a lower index as seen in equation (31). Using equation (6) the variance of an operator  $\hat{A}$  can be calculated:

$$\text{Var}(\hat{A}) = \langle \hat{A}^2 \rangle_\varphi - \langle \hat{A} \rangle_\varphi^2, \quad (32)$$

and the standard deviation:

$$\sigma = \sqrt{\text{Var}(\hat{A})} = \sqrt{\langle \hat{A}^2 \rangle_\varphi - \langle \hat{A} \rangle_\varphi^2}. \quad (33)$$

### 2.3.3. Operator Exponential

Operators can also be convoluted by exponential functions. This is particularly handy for the time evolution operator  $U(t - t_0)$  (23). Take an operator  $\hat{A}$  that can be expressed as an  $n \times n$  matrix, then its operator exponential is defined as:

$$e^{\hat{A}} = \sum_{k=0}^{\infty} \frac{\hat{A}^k}{k!} = \mathbb{1} + \hat{A} + \frac{\hat{A}^2}{2!} + \frac{\hat{A}^3}{3!} + \dots \quad (34)$$

The notation  $\hat{A}^n$  means that the operator  $\hat{A}$  is applied  $n$ -times on a state  $|*\rangle$ :

$$\hat{A}^n |*\rangle = \underbrace{\hat{A} \hat{A} \dots \hat{A}}_{n\text{-times}} |*\rangle. \quad (35)$$

If, moreover,  $A^2 = \mathbb{1}$ , the matrix exponential  $e^{i\theta A}$  can be calculated explicitly as can be neatly expressed as:

$$e^{i\theta \hat{A}} = \cos(\theta) \mathbb{1} + i \sin(\theta) \hat{A}. \quad (36)$$

A quick derivation of the equation (36) can be found in appendix (B).

Equation (36) can also be written as a matrix. For example, set  $\hat{A} = \sigma_x = X$  (see sec. 44):

$$e^{i\theta X} = \cos(\theta) \mathbb{1} + i \sin(\theta) X = \begin{bmatrix} \cos(\theta) & i \sin(\theta) \\ i \sin(\theta) & \cos(\theta) \end{bmatrix}. \quad (37)$$

In this notation, the correspondence to a rotational matrix around the x-axis is apparent. This is why, we refer to eq. (36) as a rotational operator with rotation angle  $\theta$ . We will revisit this in section 2.4.3.

## 2.4. Quantum computing

Quantum computing is computing using the phenomenons and principles of quantum mechanics. These properties are what makes quantum computing different from classical computing and where their potential lies in. For some

problems, a quantum computer can have better time complexity, even to such extremes that they are only computable on them, in feasible time scales [17]. Achieving the latter is known as **quantum advantage**. A practical quantum advantage has not been achieved yet, as the currently available quantum hardware is prone noise which limits their efficiency [18].

The fundamental information unit in quantum computing is the so-called qubit. Qubits are the quantum analogy to the classical bit. The aforementioned bit, can be either 0 or 1. A qubit, however, can be in the state  $|0\rangle$  or  $|1\rangle$  but also in any superposition of these two. These qubits can be transformed by unitary gates, some of which coincide with operators from a quantum mechanics text book such as the Pauli matrices. Quantum circuits essentially are a sequence of these gates, similar to a classical circuit.

The following section introduces qubits, prominent gates and a short exemplary circuit. Lastly, VQAs are introduced as well as their classical optimization loop.

### 2.4.1. Qubits

Qubits are two-state systems that form the fundamental information units in quantum computing, similarly to the bit in classical computing. Since a qubit can exist in a superposition of its two basis states,  $|0\rangle$  and  $|1\rangle$ , we can express an arbitrary single-qubit state as:

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \quad (38)$$

with  $\alpha, \beta \in \mathbb{C}$  and  $|\alpha|^2 + |\beta|^2 = 1$ .

The  $|0\rangle$  and  $|1\rangle$  states have already been introduced as the eigenvectors of the  $\sigma_z$  Pauli matrix, see section 2.4.2.

Since choosing a basis is arbitrary, the convention is to refer to these basis states as the computational basis states. Sometimes it is illustrative to express the complex coefficients  $\alpha$  and  $\beta$  in a different form [18]. Using  $|\alpha|^2 + |\beta|^2 = 1$  and the Pythagorean trigonometric identity, we can write the equation in terms of the angles  $\varphi$  and  $\theta$ :

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right) |0\rangle + e^{i\varphi} \sin\left(\frac{\theta}{2}\right) |1\rangle, \quad (39)$$

with  $\varphi, \theta \in \mathbb{R}$ .

By virtue of equation (39), all possible qubit states can be represented on a so-called Bloch sphere, see figure 2. Every point on this Bloch sphere is a valid qubit (pure) state, defined up to a global phase, thus giving infinite possibilities for a state representation. A mixed state<sup>4</sup> is represented on the Bloch sphere by a state within it [18].

---

<sup>4</sup>Pure states are states that can be represented by a density matrix, i.e.,  $\rho = |\psi\rangle\langle\psi|$ . Mixed states are represented by a convex sum, i.e.,  $\rho_{\text{mixed}} = \sum_i \alpha_i |\psi_i\rangle\langle\psi_i|$ .

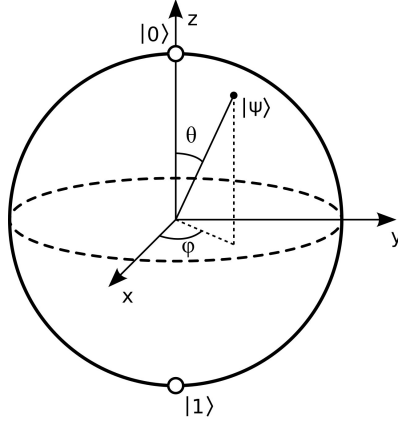


Figure 2: Bloch sphere [19]. Each axis corresponds to a pair of Pauli vectors (45). For example, the poles of the z-axis correspond to  $|0\rangle$  and  $|1\rangle$ , respectively. The angle  $\theta \in [0, \pi]$  is relative to the z-axis and the angle  $\varphi \in [0, 2\pi)$  is relative to the x-axis. Any point on and inside this sphere represents a valid state.

### Multi-qubit states

For any meaningful quantum computation, a single qubit is not sufficient. Larger qubit systems are necessary to facilitate complex quantum computations. An  $n$ -qubit quantum system is simply created by combining  $n$  individual qubits together. The corresponding mathematical operation that is used for combining such physical systems is the tensor product, e.g.:

$$|0\rangle^{\otimes n} = |0\rangle_1 \otimes |0\rangle_2 \otimes \dots \otimes |0\rangle_n = \underbrace{|0 \dots 0\rangle}_n. \quad (40)$$

For,  $n = 2$  this is called a two qubit system. Such two qubit systems have four computational basis states:

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \quad (41)$$

Any two-qubit state may be in a superposition of these four bases:

$$|\psi\rangle = \alpha |00\rangle + \beta |01\rangle + \gamma |10\rangle + \delta |11\rangle. \quad (42)$$

This state also must fulfil the normalisation condition for it to be a physically acceptable state:

$$|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1. \quad (43)$$

This can be extended to an  $n$ -qubit system analogously.

As qubits are two-state systems, similarly to spin systems,  $|0\rangle$  and  $|1\rangle$  are Pauli eigenvectors. In the next section, the eigenvectors and values of the three Pauli matrices are introduced, as they play a fundamental role in quantum computing.



### 2.4.2. Pauli Operators/ Matrices

Pauli matrices play an import role not only in spin physics but also in quantum computation. These matrices are Hermitian and unitary<sup>5</sup> and given as:

$$X = \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \quad (44)$$

Note that the Pauli matrices, together with the identity span a basis for every complex  $2 \times 2$  matrix. They all share the same two eigenvalues,  $+1$  and  $-1$ . Their corresponding eigenvectors are:

$$|+\rangle = |x_+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad |-\rangle = |x_-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad (45)$$

$$|i\rangle = |y_+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix}, \quad |-i\rangle = |y_-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -i \end{bmatrix}, \quad (46)$$

$$|0\rangle = |z_+\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = |z_-\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (47)$$

Note that the convention in the context of quantum computing is to refer to  $\sigma_x$ ,  $\sigma_y$  and  $\sigma_z$  as  $X$ ,  $Y$  and  $Y$  respectively. Correspondingly, the eigenvectors are denoted differently as well. None of these matrices commute with each other. This can be condensely expressed with a single commutation equation<sup>6</sup>:

$$[\sigma_a, \sigma_b] = 2i\epsilon_{abc} \sigma_c. \quad (48)$$

A useful way to write the Pauli matrices is to write them in terms of in terms of the computational basis, using the Dirac notation.

$$\sigma_x = |1\rangle\langle 0| + |0\rangle\langle 1|, \quad \sigma_y = i|1\rangle\langle 0| - i|0\rangle\langle 1|, \quad \sigma_z = |0\rangle\langle 0| - |1\rangle\langle 1|. \quad (49)$$

### 2.4.3. Quantum Gates

Quantum gates are unitary operators that map qubit states to qubit states. They are the elementary operations of a quantum circuit. One important difference from classical gates is the fact that quantum gates have to be reversible, since the gates are unitary and thus reversible [18].

A quantum circuit consisting of a sequence of quantum gates can be visualised by a circuit diagram. In these diagrams, time flows from left to right and multiple qubits are stacked vertically. The order in which the operators operate is, thus, also determined by this direction. For example, take one arbitrary single qubit gate and a single-qubit gate  $A$ , a circuit could look like:

$$|\alpha\rangle \text{ --- } \boxed{A} \text{ --- } |\beta\rangle$$

While gates are, in principle, not limited in the number of qubits they act on, most only act on one or two qubits. This is because every  $n$ -qubit unitary can arbitrarily well-approximated by a sequence of single- and two-qubit gates single- or two-qubit gate [18]. Moreover, there exist finite sets of quantum gates, called universal gate sets, with which it is possible to represent any

<sup>5</sup> A matrix  $A$  is called unitary if and only if:  $A^\dagger A = AA^\dagger = \mathbf{1}$ .

<sup>6</sup> with  $\epsilon_{abc}$  being the Levi-Civita symbol and using the Einstein summation convention

unitary operation up to any arbitrary precision [20]. Hence, such a universal gate set is, in principle, sufficient for all computations. Various prominent examples of single- and two-qubit gates are introduced in the following.

### Identity Gate

A trivial example for a gate is the **identity gate**. Mathematically, it is the identity matrix  $\mathbb{1}_{2 \times 2}$  and does not alter the state on which it acts. For a single qubit:

$$\mathbb{1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = |0\rangle\langle 0| + |1\rangle\langle 1|. \quad (50)$$

### Pauli Gates

The Pauli matrices also can be used as gates because they are unitary as well. All three of them rotate a given state along the corresponding axis by  $\pi$  on the Bloch sphere.

#### $X$ -gate / NOT-gate

The  $X$ -gate is also called the **bit-flip** gate because it maps  $|0\rangle$  to  $|1\rangle$  and  $|1\rangle$  to  $|0\rangle$ . It is equivalent to the NOT gate from classical circuits [21].

$X$  acts on  $|0\rangle$  and  $|1\rangle$  as follows:

$$X|0\rangle = (|1\rangle\langle 0| + |0\rangle\langle 1|)|0\rangle = |1\rangle, \quad (51)$$

$$X|1\rangle = (|1\rangle\langle 0| + |0\rangle\langle 1|)|1\rangle = |0\rangle. \quad (52)$$

#### $Y$ -gate

The  $Y$ -gate interchanges  $|0\rangle$  and  $|1\rangle$  like the  $X$ -gate except for an additional phase.

$Y$ -gate acts on  $|0\rangle$  and  $|1\rangle$  as follows:

$$Y|0\rangle = (i|1\rangle\langle 0| - i|0\rangle\langle 1|)|0\rangle = +i|1\rangle, \quad (53)$$

$$Y|1\rangle = (i|1\rangle\langle 0| - i|0\rangle\langle 1|)|1\rangle = -i|0\rangle. \quad (54)$$

#### $Z$ -gate

The  $Z$ -gate is also called the **phase flip** as it introduces a relative phase between the computational basis states

$Z$ -gate acts on  $|0\rangle$  and  $|1\rangle$  as follows:

$$Z|0\rangle = (|0\rangle\langle 0| - |1\rangle\langle 1|)|0\rangle = +|0\rangle, \quad (55)$$

$$Z|1\rangle = (|0\rangle\langle 0| - |1\rangle\langle 1|)|1\rangle = -|1\rangle. \quad (56)$$

### Rotation Gates

The previous Pauli gates can only rotate a state by a fixed angle of  $\pi$  their respective axis. To rotate a state by an arbitrary angle  $\theta$  along an axis, the **Pauli rotation** gates [21] may be used. These gates are generated by the Pauli operators as exponential, as in equation (36).

### $R_X$ -gate

The  $R_X$ -gate rotates a state along the  $x$ -axis by  $\theta$ . Using equation (36):

$$R_X(\theta) := e^{-i\frac{1}{2}\theta X} = \cos\left(\frac{\theta}{2}\right)\mathbb{1} - i\sin\left(\frac{\theta}{2}\right)X = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}.$$

In particular, setting  $\theta = \pi$  yields the  $X$ -gate up to a global phase  $-i$ :

$$R_X(\pi) := e^{i\frac{\pi}{2}}X = \begin{bmatrix} 0 & -i \\ -i & 0 \end{bmatrix} = -iX. \quad (57)$$

### $R_Y$ -gate

Analogously, the  $R_Y$ -gate rotates a state along the  $y$ -axis by  $\theta$ .

$$R_Y(\theta) := e^{-i\frac{1}{2}\theta Y} = \cos\left(\frac{\theta}{2}\right)\mathbb{1} - i\sin\left(\frac{\theta}{2}\right)Y = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}.$$

### $R_Z$ -gate

The  $R_Z$ -gate rotates a given state along the  $z$ -axis by  $\theta$ :

$$R_Z(\theta) := e^{-i\frac{1}{2}\theta Z} = \cos\left(\frac{\theta}{2}\right)\mathbb{1} - i\sin\left(\frac{\theta}{2}\right)Z = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix} \quad (58)$$

When multiple rotations are performed, they can be combined into a single gate using the **Baker–Campbell–Hausdorff formula** [18]. Which provides a closed-form solution to the equation  $e^A e^B = e^C$ , with  $A$ ,  $B$  and  $C$  operators with:

$$C = A + B + \frac{1}{2}[A, B] + \frac{1}{12}[A, [A, B]] - \frac{1}{12}[B, [A, B]] + \dots \quad (59)$$

When two rotations are performed along the same axis, for example the  $X$ -axis, the angles are simply added up. Verifying this is straight forward due to the commutation relation for the  $X$  Pauli operator (see eq. (48)), i.e.  $[X, X] = 0$ :

$$R_X(\theta)R_X(\varphi)|*\rangle = \left(e^{-i\frac{\theta}{2}X}e^{-i\frac{\varphi}{2}X}\right)|*\rangle = e^{-i\frac{\theta+\varphi}{2}X}|*\rangle = R_X(\theta + \varphi)|*\rangle, \quad (60)$$

with  $|*\rangle$  being an arbitrary single qubit state.

The circuit diagram consequently is as follows:

$$\text{---} \boxed{R_X(\theta)} \text{---} \boxed{R_X(\varphi)} \text{---} = \text{---} \boxed{R_X(\theta + \varphi)} \text{---}$$

### Hadamard gate

The so-called Hadamard gate  $H$ , creates an equal superposition of the computational basis states.

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \frac{1}{\sqrt{2}} (|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|). \quad (61)$$

In fact, this gate maps the  $|0\rangle$  to the  $|+\rangle$  state and the  $|1\rangle$  to the  $|-\rangle$ . The states  $|+\rangle$  and  $|-\rangle$  are introduced in section 2.4.2. Therefore, the Hadamard directly maps the computational basis to the Pauli  $Y$ -basis (sometimes also called the Hadamard basis).

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} |0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = |+\rangle, \quad (62)$$

$$H|1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} |1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle = |-\rangle. \quad (63)$$

Lastly, applying the  $H$ -gate twice will not alter the state. Because  $HH = \mathbb{1}_{2 \times 2}$ .

$$H(H|0\rangle) = H|+\rangle = |0\rangle, \quad (64)$$

$$H(H|1\rangle) = H|-\rangle = |1\rangle. \quad (65)$$

### CNOT-gate

In contrast to the other gates in this section, the CNOT-gate takes two qubits as its input. The first qubit acts as the control qubit, the other as the target. As the name suggest, the control qubit dictates how the target qubit is altered. The control qubit remains unaffected and the  $X$ -gate is applied to the target qubit if and only if the control qubit is in the  $|1\rangle$  state.

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (66)$$

For example, if the CNOT-gate acts on the state  $|11\rangle$ :

$$\text{CNOT} |11\rangle = |1\rangle \otimes X|1\rangle = |1\rangle \otimes |0\rangle = |10\rangle. \quad (67)$$

The full effect of the CNOT-gate is listed in table 1. Lastly, this controlled two-qubit gate operation can be generalised to the controlled- $U$  gate:

$$CU = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{bmatrix}. \quad (68)$$

This gate applies the unitary  $U$  in the same manner as the CNOT-gate applies the  $X$ -gate. A tabular summary of the previously introduced gates can be seen in appendix B.

Table 1: Table of the CNOT-gate acting on the computational bases states.

Before	After
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$
$ 10\rangle$	$ 11\rangle$
$ 11\rangle$	$ 10\rangle$

#### 2.4.4. Measurements in Different Bases

Measurements are performed in the computational basis, which - by convention - coincides with the Pauli-z basis, by default. In a circuit diagram this is represented by the according measurement symbol. For example:

$$|\psi\rangle \longrightarrow \boxed{\text{meter}} \longrightarrow \{0, 1\}$$

By the postulates of quantum mechanics, such a measurement collapses the quantum state to one of the computational basis states and produces one bit of classical information, according to the collapsed state. Therefore, before performing the measurements again, requires the re-preparation of that state. Measurements result in either the 0 or 1 bit, depending on the measured state. If an  $n$ -qubit system is measured like this qubit-by-qubit, a bit-string,  $\{0, 1\}^n$ , is produced. However, a different basis measurement can also be selected. Of particular interest are the three Pauli bases. In order to measure in either the Pauli  $X$  or  $Y$  basis in similar fashion, the state has to be transformed first. Solving this problem is a basic one in linear algebra as it essentially boils down to a basis transformation. For example, the Hadamard gate  $H$  maps the states  $|0\rangle$  and  $|1\rangle$  following eq. (62) & (63). Applying the Hadamard gate the other way around results in the wanted transformation:

$$|+\rangle \mapsto |0\rangle \quad (69)$$

$$|-\rangle \mapsto |1\rangle. \quad (70)$$

After this operation, the state  $|0\rangle$  is measured if and only if the quantum state has previously been the  $|+\rangle$ -state. As a consequence, measuring this state will yield the standard bit-string  $\{0, 1\}$  as desired.

The measurement in the Pauli  $Y$ -basis can also be achieved with an additional phase gate  $S$ . Hence, applying the right unitary transformation before measuring allows for measurements in different bases. These unitaries are listed in table 2.

Table 2: Unitary transformation for measurement in different bases.

Basis	Unitary
$Z$	$\text{---} \boxed{1} \text{---}$
$X$	$\text{---} \boxed{S^\dagger} \text{---} \boxed{H} \text{---}$
$Y$	$\text{---} \boxed{H} \text{---}$

with  $S^\dagger = (H\sqrt{X}H)^\dagger$ .

#### 2.4.5. Variational Quantum Algorithms

Variational quantum algorithms (VQA) are one of many methods that aim to utilise the currently available quantum hardware with all of its limitations. These limitations range from the limited qubit number to the limited circuit depths and general noise concerns [3, 18]. VQAs approach these limitations by using an optimization-based method. The idea is to outsource the optimization to classical computers, which interface with the quantum computer

via parameters of a parameterised quantum circuit.

VQAs may lead to a general quantum advantage on currently available NISQ hardware . The proposed use-cases for VQAs are in quantum chemistry, e.g., finding ground states, combinatorial problems, or linear algebra problems, i.e. solving large systems of linear equations [3]. A simple schematic overview of a VQA can be seen in figure 3.

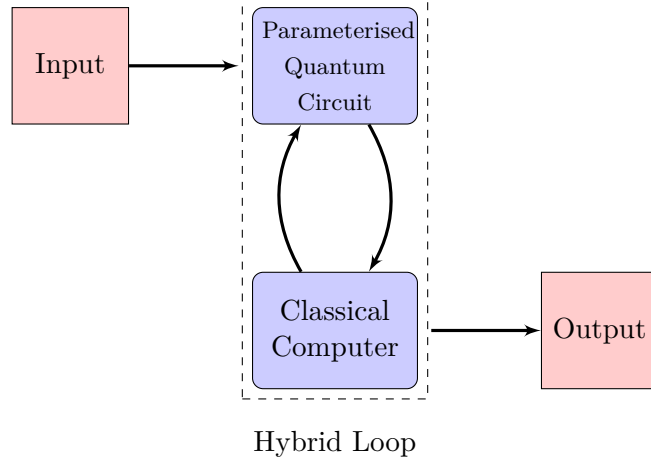


Figure 3: Simple schematic of a variational quantum algorithm (VQA). The inputs of a VQA are the Hamiltonian providing the cost function and the ansatz (the layout of the parameterised circuit). The quantum circuit prepares a quantum state to be measured repeatedly and the classical computer in return calculates the next set of parameters based on the chosen optimisation routine. The desired output in our case should correspond to the set of parameters that minimises the cost function, e.g., the ground-state of the Hamiltonian.

In order to define the problem instance for the VQA, a so-called cost function  $C$  has to be defined. The purpose of the VQA is to minimise this cost function by finding a set of parameters  $\theta$  that minimises, i.e., to ideally find .

$$\theta_{\min} = \arg \min_{\theta} C(\theta). \quad (71)$$

The cost function together with the parameter space makes up the **cost landscape**. Its dimension equals the number of parameters. A cost function, for example, could be the expectation value of the given Hamiltonian or a spin measurement. In this case, the landscape is a energy landscape. Finding a global or local minimum in this energy landscape, essentially, is the task of the VQA. The cost function in this case has the form [3]:

$$C(\theta) = \langle \psi_0 | \hat{U}^\dagger(\theta) \hat{O} \hat{U}(\theta) | \psi_0 \rangle. \quad (72)$$

with  $|\psi_0\rangle$  being the initial input state for the circuit,  $\hat{O}$  the observable (e.g. Hamiltonian),  $\hat{U}(\theta)$  a parameterised unitary, i.e., the parameterised circuit. The parameterised circuit is typically represented by multiple layers of unitaries  $\hat{U}_i(\theta_i)$  that multiply up to the whole circuit  $\hat{U}(\theta)$  as seen in Figure 4. Such unitary  $\hat{U}(\theta)$  can be expressed as:

$$\hat{U}(\theta) = \hat{U}(\theta_1) \cdot \hat{U}(\theta_2) \cdot \dots \cdot \hat{U}(\theta_L). \quad (73)$$

The explicit form of each layer depends on the ansatz made. An initial state  $|\psi_0\rangle$  with a given ansatz  $\hat{U}(\boldsymbol{\theta})$ , i.e. the circuit, results in the parameterised state  $|\psi(\boldsymbol{\theta})\rangle$ .

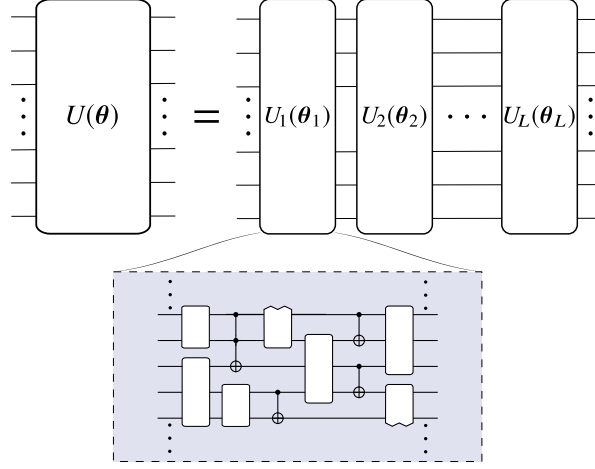


Figure 4: Parameterised circuit example circuit with  $L$  layers of unitaries. Each layer  $\hat{U}_i(\theta_i)$  depends only on a single parameter  $\theta_i$ . Stacking up these layers results in the unitary  $\hat{U}(\boldsymbol{\theta})$ . The blue box shows an example circuit for a single layer which can at most depend on  $\theta_i$ . Figure taken from Ref. [3].

The classical computer then evaluates the value of the cost function from repeated measurements and applies a classical optimisation method. The task of the classical computers is to choose the next set of parameters  $\boldsymbol{\theta}$  to navigate downwards in the cost landscape. Subsequently, the **hybrid loop** repeats. How the navigation is done depends on the method used for the optimization routine. The most commonly used is the so-called **gradient descent method**. Although there are more available methods, in this thesis, the gradient descent method is the method of choice.

## Gradient Descent

The gradient-descent method updates a set of parameters  $\boldsymbol{\theta}$  by using the gradient of the cost function with respect to  $\boldsymbol{\theta}$  such that the updated direction points to the direction in the cost landscape where the slope is the steepest. This is done by setting a learning rate  $\alpha$  and calculating the gradient at the current point  $\boldsymbol{\theta}_k$ .

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha \nabla C(\boldsymbol{\theta}_k). \quad (74)$$

Using equation (74), a classical computer calculates the next set of parameters to navigate in the cost landscape. By doing so, (local) minima can be reached. However, calculating the gradient of the cost function is not always analytically possible and one has to settle for approximative methods for the gradient estimation. A method of calculating quantum gradients is the so-called **simultaneous perturbation stochastic approximation** (SPSA) introduced in Ref. [22]. SPSA requires two circuit calls per gradient estimation, whereas the parameter shift rule scales linearly with the number of circuit parameters, making it advantageous for large circuits because of the constant scaling. The

SPSA method achieves this by estimating the cost function at  $\boldsymbol{\theta} + \mu\Delta$  and  $\boldsymbol{\theta} - \mu\Delta$ , where an element of  $\Delta$ ,  $\Delta_i$  is defined as:

$$\Delta_i = \begin{cases} +1 & \text{with probability 50\%} \\ -1 & \text{otherwise} \end{cases} \quad (75)$$

with  $i = \{1, \dots, N\}$ ,  $N$  being the number of parameters and  $\mu \in \mathbb{R}$  a small parameter.

The gradient is calculated by the following equation:

$$\nabla C(\boldsymbol{\theta}) \approx \frac{C(\boldsymbol{\theta} + \mu\Delta) - C(\boldsymbol{\theta} - \mu\Delta)}{2\mu\Delta}. \quad (76)$$

This method requires fewer samples for a gradient estimation, compared to the parameter shift rule, as it only requires a constant of 2 measurements, see Ref. [22] and appendix B.2. Especially for higher numbers of parameters, due to the constant scaling of the SPSA method, compared to the  $2n$  ( $n$  is the number of qubits) scaling of other conventional methods.

For example, the gradient descent method on a one-dimensional energy landscape, thus having only one parameter  $\theta$ , can be seen in figure 5.

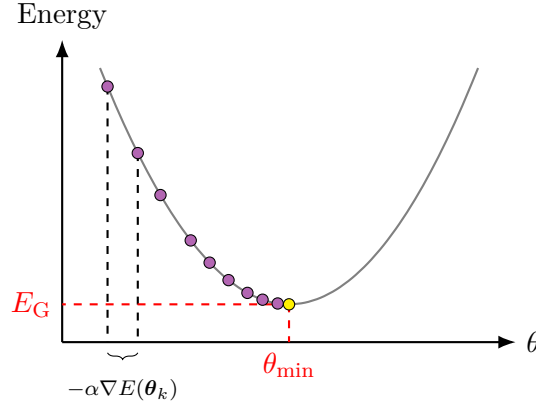


Figure 5: Gradient descent on a one-dimensional energy landscape. The yellow dot is at  $\theta_{\min}$  and bears the ground energy  $E_G$  of this energy landscape. On every purple dot, a measurement of the energy  $E(\theta_k)$  has been made. The next parameter  $\theta_{k+1}$  is calculated based on the measured gradient  $\nabla E(\theta_k)$  from the previous step. By walking stepwise through the landscape, with the right learning rate  $\alpha$ , eventually an energy  $E(\theta_k) \approx E_G$  (yellow dot) will be reached. In the region near an extremum or saddle point, that is, where the gradient is small, the number of steps to advance further becomes larger. This is shown by the purple dots being closer to each other near the yellow dot.

## Variational Quantum Eigensolvers

The most variant of VQAs is the so-called **variational quantum eigensolver** (VQE). It is used to estimate ground-state eigenvalues and eigenstates of a given Hamiltonian  $\hat{H}$  [3]. In this case, the cost function (see eq. (72)) is given by the expectation value of  $\hat{H}$  with a parameterised state  $|\psi(\boldsymbol{\theta})\rangle$ :

$$C(\boldsymbol{\theta}) = \langle \psi(\boldsymbol{\theta}) | \hat{H} | \psi(\boldsymbol{\theta}) \rangle := E(\boldsymbol{\theta}). \quad (77)$$



with  $E(\boldsymbol{\theta})$  being the energy depending on the parameters  $\boldsymbol{\theta}$ . From now on, the cost function will be called  $E(\boldsymbol{\theta})$ . Using the Rayleigh-Ritz variational method the energy  $E(\boldsymbol{\theta})$  is lower bounded by the ground state energy, i.e.

$$E_G \leq E(\boldsymbol{\theta}). \quad (78)$$

The Hamiltonian for a VQE is usually decomposed as a sum of multi-qubit Pauli operators (see sec. 2.4.2):

$$\hat{H} = \sum_i c_i P_i, \quad (79)$$

with  $c_i \in \mathbb{R}$  and  $P_i \in \{X, Y, Z, \mathbb{1}\}^{\otimes n}$ .

It is common to represent a Hamiltonian as a sum of Pauli operators, as these are easily implementable. The typical procedure for a VQE can be seen in figure 6.

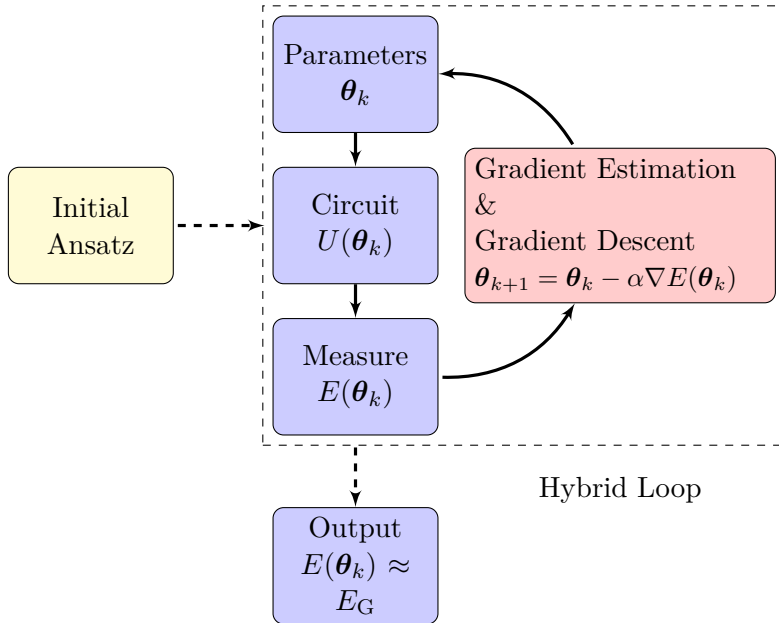


Figure 6: Flow chart of a VQE. The circuit layout is set by the Ansatz, shown in yellow. The dashed box corresponds to the hybrid loop in which the quantum circuit and the classical optimiser work together. In each loop, the circuit provides the classical optimiser the current energy  $E(\boldsymbol{\theta}_k)$  from which it calculates the next set of parameters  $\boldsymbol{\theta}_{k+1}$ . At that point, the loop continues with  $\boldsymbol{\theta}_{k+1}$  as the new starting parameters. The loop is based on figure 3.

### 3. Methods

This section provides an introduction to the empirical Bernstein stopping algorithm (EBS) and the layout of the VQE (see sec. 2.4.5) used as a benchmark. To that extent, first EBS itself and some modifications are introduced. These modified algorithms are then compared against each other and also to the Hoeffding’s bound (see eq. (15)). Secondly, we explain how to set up a VQE as a benchmarking tool for EBS. This includes, in particular, the quantum decomposition that yields a circuit representation for a unitary operator. Lastly, we illustrate on how to interconnect EBS with the hybrid loop of an VQE

#### 3.1. Empirical Bernstein Stopping (EBS)

Sampling is a resource-intensive task in quantum computing and machine learning. In the case of quantum computing, the result of a computation has to be read-off via a measurement. However, due to the fundamental nature of quantum mechanics, any measurement projects the state into a single outcome with a certain probability. Therefore, this process has to be repeated to achieve an accurate result.

The naive approach arguably is to set a number of samples that can be acquired for each estimate, this approach, however, does not provide any guarantees for the produced estimate. Instead, it would be beneficial to have algorithms, which control the sampling process based on set parameters and terminate when a fixed accuracy of the estimate is reached.

One such algorithm is the **empirical Bernstein stopping algorithm** (EBS) developed by Mnih *et al.* [15]. EBS is an adaptive stopping algorithm, as it immediately processes the samples that are drawn. In particular, it calculates the running empirical variances. By employing the empirical Bernstein inequality, eq. (19), this allows EBS to be used in cases where the variance of the underlying probability distribution is not known, i.e., most practical scenarios.

In the following section, first the concept of a  $(\epsilon, \delta)$ -stopping rule is introduced followed by the base EBS algorithm, in both a relative error and absolute error version. Secondly, an improvement to the base EBS algorithm is discussed and compared to the base version. Lastly, a comparison to a non-adaptive sample algorithm like Hoeffding’s bound is made.

##### 3.1.1. $(\epsilon, \delta)$ -Stopping Rules

A stopping rule  $\mathbf{S}$  is a condition that when met terminates an algorithm. In addition, we require a stopping rule to terminate the algorithm eventually, regardless whether the condition is met. Mathematically, this can be expressed as:

$$\mathbb{E}[T(\mathbf{S})] < \infty, \quad (80)$$

with  $T(\mathbf{S})$  being the time the stopping algorithm terminates.

This condition formally requires that the algorithm terminates in a finite amount of time. There are many ways to do so. However, the focus in this thesis is on stopping rules that are constructed from concentration inequalities. These are the so-called  $(\epsilon, \delta)$ -stopping rules.

An  $(\epsilon, \delta)$  stopping rule takes two parameters,  $\delta$  the maximal tolerated probability of failure ( $1 - \delta$  is often called confidence) and  $\epsilon$  the relative accuracy

that the returned estimate should at least have. Any stopping rule  $\mathbf{S}$  that satisfies the following condition is an  $(\epsilon, \delta)$ -stopping rule [15]:

$$\mathbb{P}[|\hat{\mu} - \mu| \leq \epsilon|\mu|] \geq 1 - \delta, \quad (81)$$

with  $\hat{\mu}$  the estimated expected value of a real-valued random variable given as the return value of an algorithm and  $\mu$  being the expected value. Any stopping rule  $\mathbf{S}$  that fulfils equation (81) ensures, with probability of at least  $1 - \delta$ , that when it terminates, the estimated expected value  $\hat{\mu}$  will only be within a relative error  $\epsilon$  of the expected value  $\mu$ :

$$|\hat{\mu} - \mu| \leq \epsilon|\mu|. \quad (82)$$

If one considers an absolute error, one simply considers the following instead:

$$\mathbb{P}[|\hat{\mu} - \mu| \leq \epsilon] \geq 1 - \delta, \quad (83)$$

$$|\hat{\mu} - \mu| \leq \epsilon. \quad (84)$$

Henceforth, if not stated otherwise, accuracies in this thesis are considered absolute.

### 3.1.2. Empirical Bernstein Stopping Rule

In the following, we demonstrate that the Bernstein inequality can be used to create an  $(\epsilon, \delta)$  stopping rule. The detailed derivation can be found in on Ref. [15].

Let  $\mathcal{F}$  be the event in which the stopping rule fails, i.e., the algorithm stops but the required accuracy is not met.

$$\mathcal{F} = \{|\hat{\mu} - \mu| \geq \epsilon\}. \quad (85)$$

Further, define  $T_{\text{failure}}$  as the time for  $\mathcal{F}$ .

As Bernstein's inequality (17), assumes the full failure provability  $\delta$  at every step  $t$ , the probability for the event  $\mathcal{F}$  can be calculated by summing up all time steps:

$$\mathbb{P}[\mathcal{F}] = \sum_{t=1}^{\infty} \mathbb{P}[\mathcal{F} \cap \{t = T_{\text{failure}}\}] \stackrel{!}{\leq} \delta. \quad (86)$$

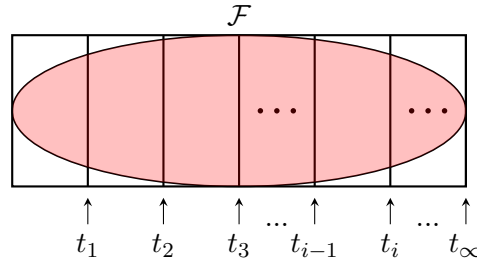


Figure 7: A more approachable representation of equation (86).  $\mathcal{F}$  is represented by the red elliptical set and the areas created by vertical lines represent disjoint subsets where  $t_i = T_{\text{Failure}}$ . The total probability of  $\mathcal{F}$  are all the times  $t_i$  the algorithm terminates but fails to deliver a return  $\hat{\mu}$  that is within  $\epsilon$  of the expected value  $\mu$ , i.e., the sum of the measures of all the subsets.

The idea is to define a sequence of partial failure probability  $\{d_t\}$ . An element  $d_t$  of this sequence can be understood as the partial failure probability for the segments in figure 7. Hence, the following inequality holds:

$$\mathbb{P}[\mathcal{F} \cap \{t = T_{\text{failure}}\}] \leq d_t. \quad (87)$$

Following equation 86, the sequence  $\{d_t\}$  must fulfil:

$$\sum_{t=1}^{\infty} d_t \leq \delta. \quad (88)$$

The author(s), Mnih *et al.* [15], recommend following choice of  $d_t$ :

$$d_t = \frac{c}{t^p}, \quad (89)$$

with  $c = \delta(p-1)/p$  and  $p > 1$  as the optimal choice.

Further discussion of this choice can be found in Ref. [15].

The empirical Bernstein bound holds for every time  $t$  (17), which serves as a useful lower-bound for each  $d_t$ :

$$\mathbb{P}[|\bar{X}_t - \mu| \geq \epsilon] \leq 2 \exp \left( -\frac{(t\epsilon)^2/2}{\sigma_t^2 + Rt\epsilon/3} \right) \leq d_t. \quad (90)$$

This equation can be rewritten into a bound of similar form as equation (18):

$$|\bar{X}_t - \mu| \leq \epsilon \leq \sqrt{\frac{2\bar{V}_t \ln \left( \frac{3}{d_t} \right)}{t}} + \frac{2R \ln \left( \frac{3}{d_t} \right)}{t} := c_t. \quad (91)$$

This equation holds with probability of at least  $1 - d_t$ .

Compared to the empirical Bernstein bound, see eq. (19), eq. 91 assumes a partial failure probability  $d_t$  instead of the full failure probability  $\delta$ . The new sequence  $\{c_t\}$  is constructed in such a way that at any time step  $t$  an element of the sequence  $c_t$  is half the width of a  $1 - d_t$  confidence interval [23].

### 3.1.3. Base Algorithm

The base form of the empirical Bernstein stopping algorithms (EBS), called **EBStopSimple**, uses the previously introduced  $c_t$  as an upper bound an accuracy  $\epsilon$ , i.e., the deviation of the empirical mean from the actual mean. Henceforth, EBS will refer to any empirical Bernstein stopping algorithm, including modifications.

The derivations above lack a stopping condition, the goal now is to use this sequence  $c_t$  (91) to construct one. Further details can be found in the Ref. [15]. Assume that the algorithm stops at a time  $t$  if:

$$c_t \leq \epsilon(|\bar{X}_t| - c_t), \quad (92)$$

then the following also holds:

$$||\bar{X}_t| - |\mu|| \leq c_t \leq \epsilon(|\bar{X}_t| - c_t) \leq \epsilon. \quad (93)$$

In the equation above, one can see that if the algorithm stops at the condition in equation (92) the estimated  $\bar{X}_t$  will be within a relative error  $\epsilon$  of the real

expected value  $\mu$ . The last inequality in equation (92) results in the stopping criterion used for the EBS algorithm:

$$c_t \leq \frac{\epsilon}{1 + \epsilon} |\bar{X}_t| \leq \epsilon |\bar{X}_t|. \quad (94)$$

The following pseudocode algorithm uses the stopping criterion (94).

---

**Algorithm 1** EBStopSimple

---

```

 $c_0 \leftarrow \infty$ 
 $t \leftarrow 1$ 
while  $c_t > \epsilon |\bar{X}_t|$  do                                ▷ equation (94)
    Sample  $X_t$ 
    Update  $\bar{X}_t$ 
    Update  $\bar{V}_t$ 
    Update  $c_t$                                             ▷ equation (91)
     $t \leftarrow t + 1$ 
end while
 $T_{\text{Stop}} \leftarrow t$                                 ▷ Time at which the algorithm terminates
 $\hat{\mu} \leftarrow \bar{X}_{T_{\text{Stop}}}$ 
return  $\hat{\mu}$ 

```

---

This algorithm stops when the current expected value  $\bar{X}_t$  is within  $\epsilon$  of the bound  $c_t$ . Therefore, due to equation (93), the algorithm can be terminated and produce a  $(\epsilon, \delta)$ -estimate. To convert the algorithm 1 from relative to absolute accuracy, simply use equation (83) as the stopping condition. The algorithm 2 is an adaptation of the EBSimpleStop algorithm that Mnih [23] proposed using an absolute accuracy.

---

**Algorithm 2** EBA

---

```

 $c_0 \leftarrow \infty$ 
 $t \leftarrow 1$ 
while  $c_t > \epsilon$  do
    Sample  $X_t$ 
    Update  $\bar{X}_t$ 
    Update  $\bar{V}_t$ 
    Update  $c_t$                                             ▷ equation (91)
     $t \leftarrow t + 1$ 
end while
 $T_{\text{Stop}} \leftarrow t$                                 ▷ Time at which the algorithm terminates
 $\hat{\mu} \leftarrow \bar{X}_{T_{\text{Stop}}}$ 
return  $\hat{\mu}$ 

```

---

This new algorithm is called EBA, the 'A' represents for the absolute accuracy. It behaves identically to the EBStopSimple algorithm, except that now an absolute accuracy is used.

If, for example, one estimates  $\mu = 2$  and sets  $\epsilon = 0.1$ , the resulting absolute accuracy would be  $\pm 0.1$ . Whereas the relative accuracy depends on the mean, i.e.  $\epsilon|\mu| = \pm 0.2$ . Importantly, a relative accuracy can cause EBS to not terminate in the case where  $\mu = 0$ . If  $|\mu|\epsilon \rightarrow 0$  then EBS has to sample infinitely many times to achieve an relative accuracy of  $\pm 0$ , i.e., a perfect

estimate, resulting in EBS not terminating.

In general, it is wasteful to check the condition of the algorithm at every time step  $t$ , this is because if the algorithm does not converge at time step  $t$  it is unlikely that it converges close to it. A method that uses this idea is content of the next section.

### 3.1.4. Modifications on EBA

Like previously mentioned, it is a resource-draining method to check the condition every time  $t$ , because if the condition is not met after  $t$  samples, it is unlikely that it is met after  $t+1$  samples. A logarithmic approach is a natural ansatz to this problem, due to the  $\mathcal{O}(1/\sqrt{t})$  scaling of  $c_t$ .

The idea is to collect a number of samples in a batch and then update  $c_t$ . Collecting samples in a batch also allows the empirical variance  $\bar{V}_t$  to further converge to the actual variance, leading to a tighter bound of  $c_t$  and thus a faster stopping time.

To this extent Mnih *et al.* [15] introduces the variables  $k$  and  $t_k$ . Although the variable  $t$  still refers to the amount of samples that have been taken,  $k$  is the number of times  $c_t$  has been updated. The idea is to delay the update of  $c_t$  such that the batching of the samples can lead to a lower bound and therefore to earlier stopping. While there are several options of how to explicitly implement this, we only consider the so-called geometric sampling method. As mentioned above, this method batches  $t_k = \lceil \beta^k \rceil$  many samples and checks the stop conditions after the  $k$ -th time. Geometric sampling checks the condition at most  $\log_\beta(t)$  times, with  $t$  being the number of samples taken. Inserting  $\log_\beta(t)$  into  $d_t$  (89) gives:

$$d_t = \frac{c}{t^p} \rightarrow \frac{c}{(\log_\beta(t))^p}. \quad (95)$$

Considering that  $c_t$  (see eq. (91)) is  $\propto \log(1/d_t)$ , a tighter bound of  $c_t$  is achieved by maximising  $d_t$ . It is clear that if  $t \rightarrow \infty$  then  $1/t^p < 1/(\log_\beta(t))^p$ , hence geometric sampling gives a tighter bound. As a consequence, the tighter bound  $c_t$  leads to earlier stopping, i.e. fewer samples for an estimate.

The bound  $c_t$  must also be modified to benefit from geometric sampling:

$$c_{t_k} = \sqrt{\frac{2\bar{V}_t \ln\left(\frac{3}{d_k}\right)}{t}} + \frac{2R \ln\left(\frac{3}{d_k}\right)}{t}. \quad (96)$$

Note that the index for  $d_k$  changed compared to equation (91), as this term is updated now with exponentially increasing gaps. Algorithm 3 below is a modification of the algorithm 2.

---

**Algorithm 3** EBAGeo: geometric sampling

---

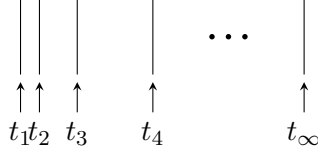
```

 $t \leftarrow 1$ 
 $k \leftarrow 0$ 
while  $c_{t_k} < \epsilon$  do
    Sample  $X_t$ 
    Update  $\bar{X}_t$ 
    Update  $\bar{V}_t$ 
     $t \leftarrow t + 1$ 
    if  $t > \lceil \beta^k \rceil$  then
         $k \leftarrow k + 1$ 
        Update  $c_{t_k}$  ▷ equation (96)
    end if
end while
 $T_{\text{Stop}} \leftarrow t$  ▷ Time at which the algorithm terminates
 $\hat{\mu} \leftarrow \bar{X}_{T_{\text{Stop}}}$ 
return  $\hat{\mu}$ 

```

---

Figure 8: Visual presentation of  $t_k = \lceil \beta^k \rceil$ . The distance between  $t_k$  and  $t_{k+1}$  becomes exponentially larger.



Geometric sampling introduces another parameter  $\beta$ , which to some degree controls the stopping time of algorithm 3. Larger  $\beta$  tend to overshoot the lowest possible stopping time up to the extent that the proposed stopping time does not take advantage of the variance information anymore. This is because the higher  $k$  becomes, the larger the difference is between  $\lceil \beta^k \rceil$  and  $\lceil \beta^{k+1} \rceil$ . While this method reduces the stopping time, it also reduces the chance to accurately hit the smallest valid stopping time  $T_{\min}$  without being off by a factor of  $\beta$ .

Hence, the authors, introduce a modification to this problem in Ref. [23], so-called **mid-interval sampling**. Using this modification, the algorithm checks the condition additionally in between  $\lceil \beta^k \rceil + 1$  and  $\lceil \beta^{k+1} \rceil$ .

---

**Algorithm 4** EBAGeoMarg: martingale-based anytime stopping

---

```
 $t \leftarrow 1$ 
 $k \leftarrow 0$ 
while  $c_t < \epsilon$  do
  Sample  $X_t$ 
  Update  $\bar{X}_t$  &  $\bar{V}_t$ 
   $t \leftarrow t + 1$ 
  if  $t > \lfloor \beta^k \rfloor$  then
     $k \leftarrow k + 1$ 
     $\alpha \leftarrow \lfloor \beta^k \rfloor / \lfloor \beta^{k-1} \rfloor$ 
     $x \leftarrow -\alpha \ln(d_k/3)$ 
     $c_{tk} \leftarrow \sqrt{2\bar{V}_t x/t} + 3Rx/t$ 
  end if
end while
 $T_{\text{Stop}} \leftarrow t$   $\triangleright$  Time at which the algorithm terminates
 $\hat{\mu} \leftarrow \bar{X}_{T_{\text{Stop}}}$ 
return  $\hat{\mu}$ 
```

---

Furthermore, the Python implementations of the previously introduced algorithms may be found in the GitHub repository **EmpiricalBernsteinAlgorithm** [24].

### 3.1.5. Effect of the Variance on EBS

The empirical Bernstein stopping algorithms generally perform better in low-variance scenarios, as shown in figure 2.2.4, where the underlying bounds are compared. In order to analyse the effect of the variance, the variables  $\epsilon$ ,  $\delta$  and the range  $R$  are kept constant while the variance is variable. One method of keeping  $R$  constant while varying the variance is to group a  $l$  random variables together, i.e., to consider their mean outcome as a new random variable. In the case of a uniformly distributed random variable, the variance of the grouped random variable thus scales with  $1/l$ .

Let  $\gamma_l$  be the mean of  $l$   $\text{uniform}(a, b)$  i.i.d. random variables  $X_i$ , i.e.

$$\gamma_l = \frac{1}{l} \sum_{i=1}^l X_i, \quad (97)$$

their variance then is as follows,

$$\sigma^2(\gamma_l) = \frac{(b-a)^2}{12l}. \quad (98)$$

Equation (98) shows that with increasing  $l$ , the variance is decreasing. Doing so allows us to create a benchmark of EBS while keeping  $\epsilon$ ,  $\delta$  and the Range  $R$  constant while decreasing the variance. A benchmark of EBA, EBAGeo, EBAGeoMarg and Höfdding's bound can be seen in figure 9.



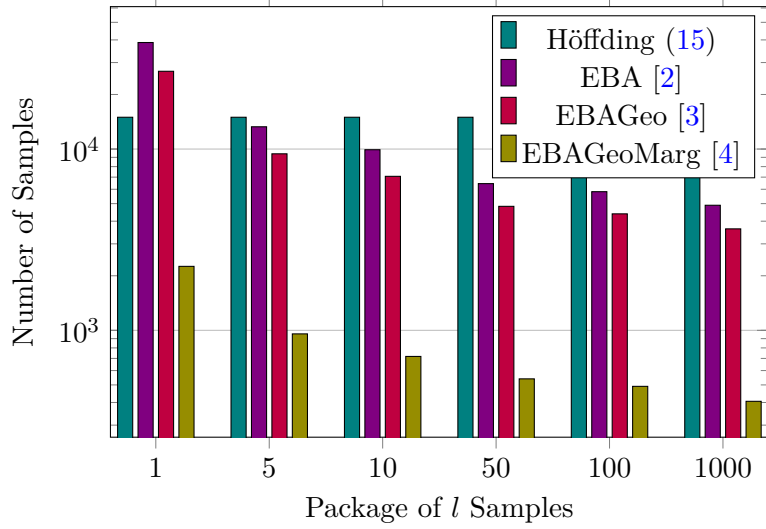


Figure 9: Effect of the variance on various sampling algorithms. Random variables are sampled from an  $\text{uniform}(a, b, l)$  distribution, with  $l$  being the number of samples that have been batched together. Each bar represents the average over 100 runs. Furthermore,  $\beta = 1.1$  is set for EBAGeo, while  $\delta = 0.1$  and  $\epsilon = 0.01$  are fixed for all algorithms. Due to eq. (98), the variance decreases with  $l$ . However, Höfding’s bound is constant for a set  $R$  as it does not depend on the variance at all.

Figure 9 demonstrates how EBS, in general, needs fewer samples as the underlying variance of the random variables gets smaller. While Höfding’s bound always yields a constant amount of samples, the EBS-based algorithms lead to a decrease of the amount of samples. It is also clear that the EBAGeoMarg algorithm outperforms the other two EBS variants. Furthermore, it seems like EBAGeo overshoot the actually stopping time by a factor  $\approx \beta^k$ , as discussed in section 3.1.4, because of the stark contrast to EBAGeoMarg.

Even though figure 9 shows a clear decrease of samples taken by the algorithms, the construction of  $\gamma_l$  increases the total number of samples taken by a factor of  $l$ . For a fair comparison, figure 10 shows the same bar charts as in Figure 9 but with the total number of samples taken adjusted by  $l$  accordingly. It shows how higher values of  $l$  lead to higher total number of samples taken. This is the case because the additional number of samples required to batch  $l$  samples into one is higher than EBS can save by a decreased variance by  $1/l$ .

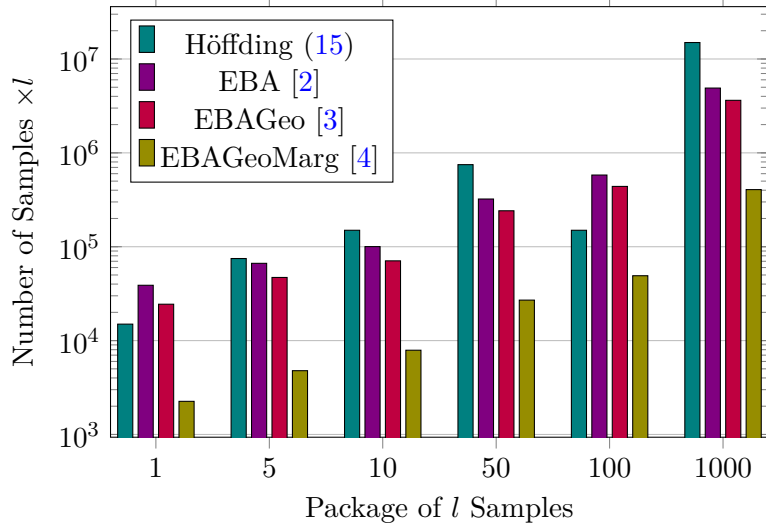


Figure 10: Data from figure 9 but adjusted to the batch size  $l$ . This represents the actual number of samples that were used. Höfding’s bound does not depend on variance, it requires more samples with increased package sizes  $l$ .

### 3.2. Constructing a suitable VQE ansatz

An issue concerning VQE is that finding an ansatz is usually a difficult problem. When working with small numbers of qubits, one can find an ansatz by using an algorithm that decomposes a unitary matrix into a sequence of quantum gates from the universal gate set only. Such a **quantum decomposition algorithm** decomposes a unitary  $2^n \times 2^n$  matrix into a sequence of  $X$ -gates and fully controlled rotational gates that act on  $n$  qubits, resulting in a circuit representation of that matrix [25]. The latter form a universal set of quantum gates, which means that any unitary gate can be expressed in terms of that set of gates.

These methods will be used in section 4.1 to construct a toy example from a small Hamiltonian, for the purpose of creating benchmarks of EBA in a VQE setting. Lastly, a quick overview of how EBA is integrated into a VQE is presented.

#### 3.2.1. Quantum Decomposition

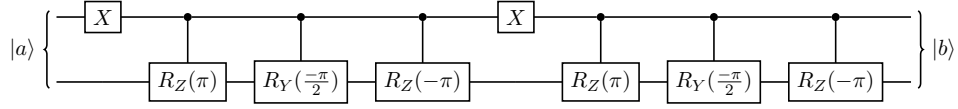
Let  $\hat{U} \in \mathbb{C}^{2n \times 2n}$  be a unitary, i.e.  $\hat{U}^\dagger = \hat{U}^{-1}$ , complex-valued matrix. Such a unitary  $\hat{U}$  can be decomposed into a combination of  $X$ -gates and fully controlled rational gates, using the quantum decomposition algorithm proposed by Fedoriaka [25]. This decomposition is computationally expensive, as it scales exponentially with the number of qubit  $n$ , i.e.,  $\mathcal{O}(4^n)$ . A Python implementation of this decomposition algorithm can be accessed via the GitHub repository in Ref. [26].

For example, let

$$\hat{U} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}, \quad (99)$$

be a unitary matrix  $\in \mathbb{C}^{4 \times 4}$ .

Using the decomposition algorithm on  $\hat{U}$  gives the following circuit:



with  $|a\rangle$  and  $|b\rangle$  being arbitrary states  $\in \mathbb{C}^4$ .

The circuit above is equivalent to the equation:  $\hat{U} |a\rangle = |b\rangle$ .

### 3.2.2. Hamiltonian to Parametrized Circuit

We want to employ a quantum circuit to find the eigenvalues of a given Hamiltonian. In the following, using the quantum decomposition algorithm, we show how to design the right circuit for this task. Usually, the input state for a  $n$ -qubit circuit corresponds to a bit string  $b \in \{0, 1\}^n$ , typically the state  $|0\rangle^{\otimes n}$ . In order to yield the eigenvalues of the Hamiltonian, the circuit  $\hat{U}$  should have this basic property:

$$\{|b\rangle\} \xrightarrow{\hat{U}} \{|\varphi\rangle\}$$

with  $\{|\varphi\rangle\}$  being eigenstates of the Hamiltonian.

Now suppose a Hamiltonian  $\hat{H} \in \mathbb{C}^{n \times n}$ . Its eigenvalue decomposition is given as:

$$\hat{H} = \hat{U} \Lambda \hat{U}^\dagger, \quad (100)$$

with  $\hat{U}$  being a matrix with the eigenvectors of  $\hat{H}$  as its columns and  $\Lambda$  being a matrix with its eigenvalues on its diagonal. This provides a recipe on how to obtain the eigenvalues from the quantum circuit from states corresponding to bit strings alone:

$$\lambda_i = \langle \psi_i | \hat{H} | \psi_i \rangle = \underbrace{\langle \psi_i | \hat{U}}_{b \in \{0,1\}^n} \underbrace{\Lambda \hat{U}^\dagger | \psi_i \rangle}_{b \in \{0,1\}^n} \stackrel{!}{=} \langle b | \Lambda | b \rangle, \quad (101)$$

with  $\{|\psi_i\rangle\}$  being eigenstates of  $\hat{H}$  and  $\{\lambda\}$  being the corresponding eigenvalue. In other words, the unitary  $\hat{U}$  is chosen such that it maps a computational basis state  $\{|b\rangle\}$ , to an eigenstate  $\{|\psi\rangle\}$  or  $\hat{U}\{|b\rangle\} = \{|\psi\rangle\}$ . This unitary  $\hat{U}$  can be decomposed into a quantum circuit via the quantum decomposition algorithm, see sec. 3.2.1. The next step is to parameterise this circuit  $\hat{U}$  by adding rotational gates to the circuit. While one naturally knows the optimal parameters beforehand using this method, it nonetheless represents a practical ansatz.

$$\begin{aligned} \{|b_i\rangle\} &\xrightarrow{\text{R}(\boldsymbol{\theta})} \xrightarrow{\hat{U}} \{|\psi_i(\boldsymbol{\theta})\rangle\} \\ \{|b_i\rangle\} &\xrightarrow{\hat{U}(\boldsymbol{\theta})} \{|\psi_i(\boldsymbol{\theta})\rangle\} \end{aligned}$$

with the gate  $\text{R}(\boldsymbol{\theta})$  representing a black box of rotational gates acting on different qubits and  $\hat{U}(\boldsymbol{\theta}) := \text{R}(\boldsymbol{\theta})\hat{U}$ .

The state  $|\psi_i(\boldsymbol{\theta})\rangle$  has the energy of  $\lambda_i(\boldsymbol{\theta})$ . By design, there exists a set of parameters  $\boldsymbol{\theta}_{\text{opt}}$  such that:

$$|\psi_i(\boldsymbol{\theta}_{\text{opt}})\rangle = \lambda_i(\boldsymbol{\theta}_{\text{opt}}) = \lambda_i. \quad (102)$$

Using these results, one can create a parameterised circuit as a circuit ansatz in order to estimate the ground state energy of the Hamiltonian. As mentioned above, this method is only feasible with small qubit numbers  $n$ , as the quantum decomposition algorithm scales exponentially with  $n$ . However, for the purpose of creating toy examples, it is satisfying to use two qubit systems.

### 3.2.3. EBA inside a VQE

In this thesis, our VQEs are using the gradient descent method and the SPSA method for approximating the gradient of the cost function. Hence, multiple energy measurements are required to:

- accurately estimate the ground state and
- approximate the gradient via SPSA.

For all the aforementioned energy measurements, EBA is used. More specifically, the **EBAGeoMarg** algorithm (see alg. 4) variation, as it is the best performing across all parameters choices (see section 3.1.5 and Ref. [23]). A schematic overview of how the VQE will work can be seen in figure 11.

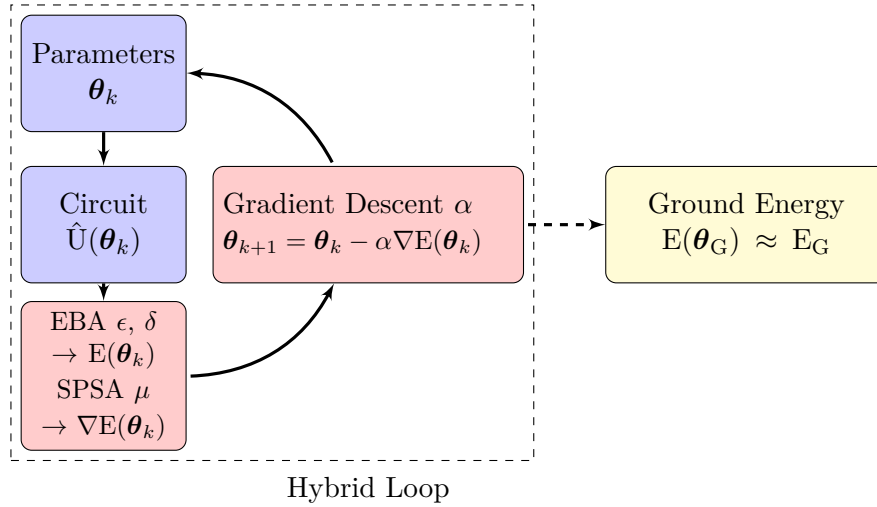


Figure 11: Abbreviated version of the flow chart 6 for a VQE. We omit the ansatz, see section 3.2. The blue boxes are part of the quantum computer, while the red part is part of the classical computer. The dashed box corresponds to the hybrid loop, in which the quantum circuit and the classical optimiser work together. In each loop, the circuit provides the classical optimiser with the current energy  $E(\theta_k)$  from which it calculates the next set of parameters  $\theta_{k+1}$ . The energy  $E(\theta_k)$  is estimated using EBA, while the gradient  $\nabla E(\theta_k)$  is estimated using the SPSA method. Using these measurements, the loop restarts with  $\theta_{k+1}$  which are calculated using the gradient descent method. The Yellow box shows the estimated ground energy  $E(\theta_G)$ . The loop is based on figure 3.

## 4. Results

In the following, the results of the benchmarks are presented. Firstly, a VQE toy example is created using the procedure from the previous section. Based on this toy example, the effect of the parameter  $\epsilon$  on accuracy and the number of samples is studied. In addition to the EBA algorithm, the Höfdding's bound is also used for further comparisons.

Secondly, EBA is used to estimate the ground state energy of a  $H_2$  molecule for varying bond lengths  $d$ . This results in a dissociation curve of  $H_2$  based on the distance of the two hydrogen atoms. The focus of this experiment is on reliably reaching the necessary accuracy for these kinds of estimates.

Henceforth, we fix the EBSGeoMarg related parameters  $p = 1.1$  and  $\beta = 1.1$ . Moreover, EBSGeoMarg alg. [4] is the EBS algorithm of choice. Furthermore, the confidence  $1 - \delta$  will not be analyzed in this thesis as one can always achieve a higher confidence by repeating the experiment [14]. Henceforth, confidence is set to  $1 - \delta = 0.9 \equiv 90\%$ .

### 4.1. Toy Example: Two Qubit System

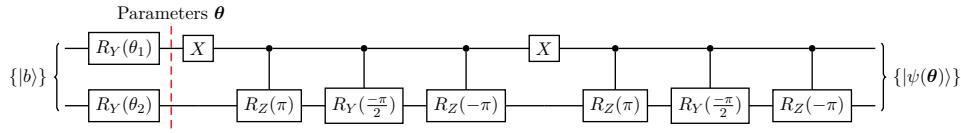
Consider the Hamiltonian:

$$\hat{H} = (Z \otimes \mathbb{1}) + (\mathbb{1} \otimes X). \quad (103)$$

This explicit Hamiltonian is chosen because it acts on two qubits and is therefore small enough to be used in the quantum decomposition algorithm. The reason being that using this algorithm,  $\hat{H}$  can be represented as a quantum circuit, and subsequently be parameterised. Hence, if used in a VQE, it is guaranteed that the solution/ground state is contained within it. Doing so allows to faithfully construct a VQE for benchmark purposes.

Additionally, such Hamiltonian can easily be analyzed, i.e. eigenvector and eigenvalues, making it straightforward to verify results. The explicit matrix form of  $\hat{H}$ , its eigenstates and eigenvalues, can be seen in the Appendix C.

This Hamiltonian acts on two qubits and has the eigenstates  $\{|\psi\rangle\}$  with the respective eigenvalues  $\{|\lambda\rangle\}$ . Applying the methods of section 3.2 on  $\hat{H}$  or more specifically on its eigenvector matrix  $\hat{U}$  results in the example circuit in section 3.2.1. The parameterised circuit based on  $\hat{U}$  can be seen below.



with  $\theta = (\theta_1, \theta_2)$ .

This is only one of the many possible ways to parameterise a circuit. Because the rotational gates have a periodicity of  $4\pi$  a highly symmetric behaviour of  $\theta_1$  and  $\theta_2$  is expected. In this particular choice of parameterization, the optimal parameters are  $\theta = (3n\pi, 3m\pi)$  with  $n, m \in \mathbb{N}_{>0}$ .

Using this circuit, one can construct a VQE by simply optimizing the parameters  $\theta_1$  and  $\theta_2$ .

The ground state energy of this Hamiltonian is  $E_G = -2$  with the eigenstate  $|\psi_G\rangle = 1/\sqrt{2}(|11\rangle - |10\rangle)$  (see appendix C). This eigenstate and therefore this

eigenvalue is what the VQE should converge towards. Additionally, it does not matter what bit string  $\{|b\rangle\}$  is put into the parameterised circuit, as this only affects the position of the ground state. Nonetheless, the state  $|00\rangle$  is used as the initial state.

It is beneficial to visualise, e.g., the expected energy as a function of the parameters. The expected energy is given by,  $\langle\psi(\boldsymbol{\theta})|\hat{H}|\psi(\boldsymbol{\theta})\rangle$ . The advantage of having an efficiently solvable Hamiltonian is that one can calculate, e.g., the expected energy analytically exact.

For instance, in the figure 12 the expected energy and variance of  $\hat{H}$  are displayed as a function of  $\boldsymbol{\theta}$  (both with respect to  $|00\rangle$ ). Low variance regions exist near the minima/maxima, while far from critical points the variance is high. In those parameter regions of low variance, we expect EBS to significantly require fewer samples for a given accuracy, see sec. 3.1.5.

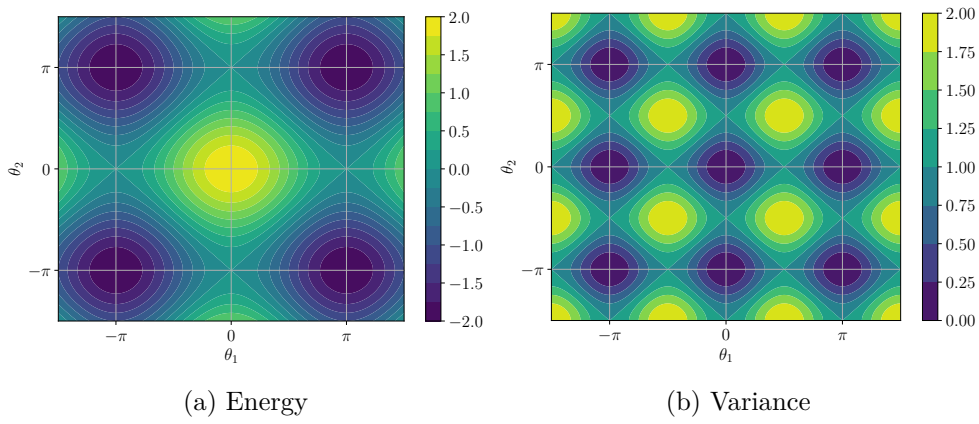


Figure 12: Expected energy (a) and the variance (b) of  $\hat{H} = (Z \otimes \mathbb{1}) + (\mathbb{1} \otimes X)$  with respect to  $\hat{U}(\boldsymbol{\theta})|00\rangle$  as a function of the parameters  $\boldsymbol{\theta} = (\theta_1, \theta_2)$ . Both plots are plotted in an interval of  $\theta_{1,2} \in [-\frac{3}{2}\pi, \frac{3}{2}\pi]$ . The variance is exactly zero at each minima/maxima and saddle point, while being maximal between them.

The gradient descent hyperparameters, i.e. the learning rate  $\alpha$  and step size  $\mu$ , do not have a drastic effect on the overall convergence of the VQE, due to the highly structured landscape of the chosen Hamiltonian  $\hat{H}$ .

A run of the VQE refers to a sequence of energy estimations (using EBS) terminating at the desired ground state energy, in this instance  $E_G = -2$ .

At each step in the sequence, the algorithm determines the energy, its derivative and updates the parameters according to the gradient descent algorithm, see sec. 2.4.5. A collection of runs is embedded in the energy and variance landscapes, respectively, can be found in figure 13. Further, each VQE initialises to  $\boldsymbol{\theta} = (0, 0)$  and terminates if,  $|\hat{E} - E| \leq 0.2$  is fulfilled. The jagged motion that is observed is due to the SPSA method for the gradient descent. SPSA allows an update only in  $2^L$  direction, with  $L$  being the number of parameters, hence here only 4 directions for an update are possible.

Close to a minima/maxima or saddle point the variance goes to zero causing EBS to require fewer samples for a prediction with the same accuracy level. We checked that the number of samples required by EBS as a function of either variance or distance from the origin exhibits this expected behaviour (data not shown).

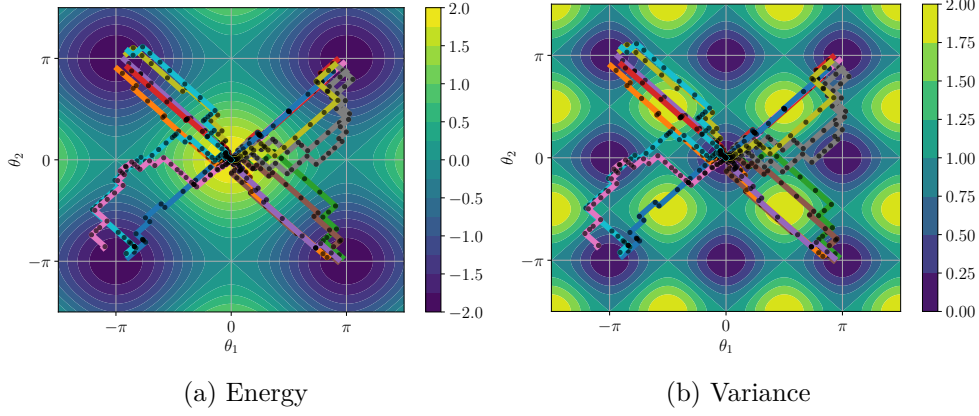


Figure 13: Same plots as fig. 12 with multiple runs of the VQE embedded on top. Each colour represents a different run while the black circles represent an update step on that run. In total, there are 11 distinctive runs displayed. These runs were created using the following parameters:  $\alpha = 0.25$ ,  $\mu = 0.1$ ,  $\epsilon = 0.2$  and  $\delta = 0.1$ . The seemingly jagged motion of the path, is due to SPSA method. The high symmetric nature of the energy landscape renders each of the four surrounding minima equally likely to be converged to.

Further, one can gauge the overlap of the estimated state with the actual ground state, using the fidelity  $F$ . The fidelity is defined as the absolute value of the inner product of two states  $|\phi\rangle$  and  $|\varphi\rangle$ , that is:

$$F(\psi, \varphi) = \|\langle\psi|\varphi\rangle\|. \quad (104)$$

By design, the fidelity is one if the states are the same and zero if they are orthogonal. Therefore, if the fidelity is plotted over the iterations of a VQE, it is expected to converge to 1 the same time the algorithm converges to the ground energy. To this extent, in the figure 14, the fidelity  $F(\psi_G, \psi(\boldsymbol{\theta}))$  and energy as a function of the number of iterations are plotted.

It confirms the convergence to the correct state, i.e., as  $F(\psi_G, \psi(\boldsymbol{\theta}))$  converges to one, the energy converges to  $-2$ , indicating the correct estimation of the ground energy (and state). The fidelity of the final VQE state and the actual ground state is approximately  $F(\psi_G, \psi(\boldsymbol{\theta}_{opt})) \approx 0.995$ .

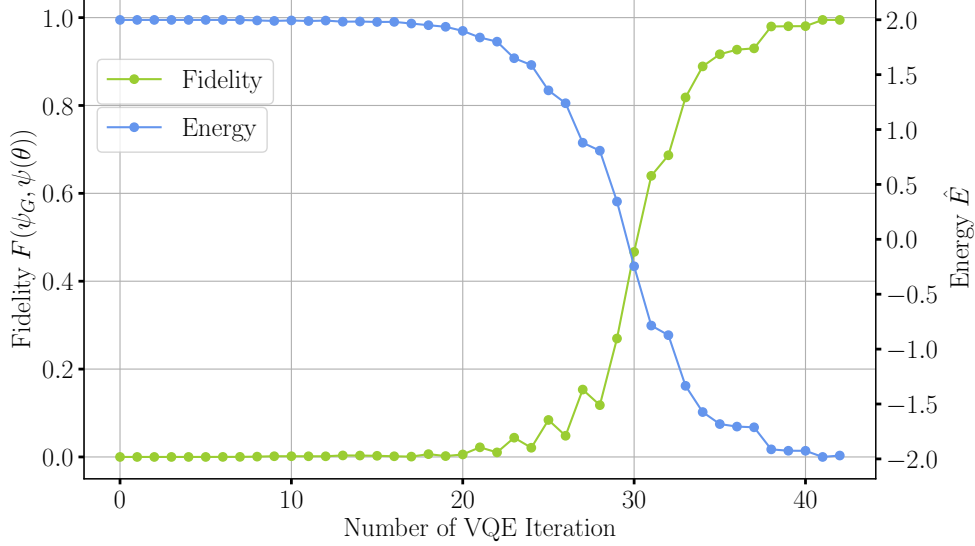


Figure 14: This figure shows a plot of the fidelity  $F(\psi_G, \psi(\theta))$  and the estimated energy  $\hat{E}$  as a function of the number of VQE update iterations. A dot represents the fidelity or energy, while the lines connecting the dots are only included for visual guidance. While the Fidelity converges to one, the energy converges to -2, i.e. the ground state energy.

#### 4.1.1. Effect of $\epsilon$ on Actual Accuracy and Success Probability

The parameter  $\epsilon$  has the strongest influence on the behaviour of EBS. Therefore, analysing it is crucial for any application of EBS.  $\epsilon$ , in this case, is the absolute error or accuracy that the  $(\epsilon, \delta)$ -estimated of EBS has. A lower value of  $\epsilon$ , meaning higher accuracy, results in EBS requiring more samples to attain said accuracy. However, during our analysis, we noticed that EBS usually produced estimates with a higher accuracy than the minimum required accuracy  $\epsilon$ .

This can be seen in figure 15, where we show the actual accuracies observed during the VQE optimization for various accuracies  $\epsilon$  with which EBS has been initialised. To that extend, for each  $\epsilon$  100 VQE runs are conducted and displayed together.

They show that EBS effectively returns an estimate of higher accuracy than the required level  $\epsilon$ . Moreover, this effect is more pronounced for higher accuracies. In addition to this higher measured accuracy  $\tilde{\epsilon}$ , one can see that even though the confidence probability  $1 - \delta = 90\%$  is relatively high, the resulting empirically measured confidence is usually higher. This effect is increased for higher values of  $\epsilon$ .



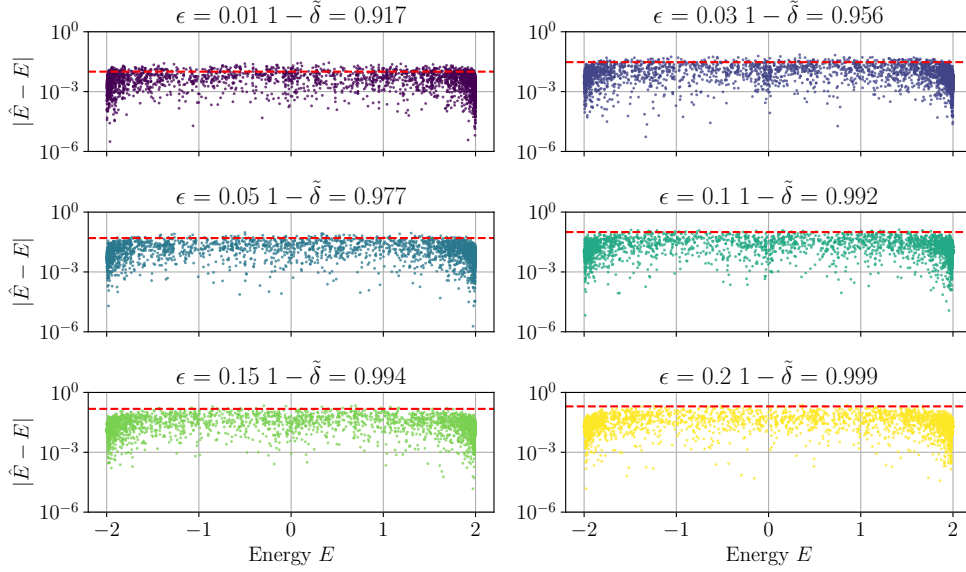


Figure 15: Distance/ actual accuracy  $|\hat{E} - E|$  as a function of the analytical energy  $E$  for different values of  $\epsilon$ , with  $\hat{E}$  being the estimated energy. The red dashed line is the corresponding value of  $\epsilon$ . Each dot is a single step across 100 runs of the VQE.  $1 - \tilde{\delta}$  refers to the measured confidence, i.e.,  $\mathbb{P}[|\hat{E} - E| \leq \epsilon]$ . Following parameters for the VQE were used:  $\alpha = 0.25$ ,  $\mu = 0.1$  and  $\delta = 0.1$ . The actual accuracy  $|\hat{E} - E|$  is usually much below the required accuracy level  $\epsilon$  (red dashed line). While there are estimates that are above the red line, for all  $\epsilon$ , this is within the set  $1 - \delta$  confidence. Close to the eigenvalues 2 and  $-2$  (energy of the initial parameters/ ground-state energy), the accuracy increases significantly, as the variance there vanishes.

In figure 15 only a few data energy measurements fail to reach the required accuracy and, thus, lie above the red dashed line. However, these estimates that are not within  $\epsilon$  of the actual energy value are well within confidence  $1 - \delta$ . These higher measured accuracies are due to EBS only guaranteeing that an estimate deviates at-least  $\epsilon$  from its actual mean.

The influence of the parameter  $\delta$  is not analysed here, as one can always achieve a higher confidence by repeating the experiment  $n$  times. This leads to a decrease of  $\delta$  as the potential failures get exponentially decreased [14].

#### 4.1.2. Effect of $\epsilon$ on Sample Complexity

Increasing the accuracy  $\epsilon$  naturally requires more samples to achieve that accuracy. In settings where a certain accuracy is required, but the resources are limited, e.g., hybrid quantum algorithms, reducing the amount of samples while keeping the accuracy high is crucial. From a practical standpoint, arbitrary amounts of measurement rounds are just not feasible. Hence, a sample limit of  $10^5$  set. A sample limit is a simple method to structure the sampling process; if it is reached, it can be assumed that the algorithm does not converge to the ground state. However, the EBAGeoMarg algorithm does not reach this sample limit for the values chosen for  $\epsilon$  in this benchmark. We visualise the dependency between empirical variance and the resulting number

of samples taken by EBS in fig. 16.

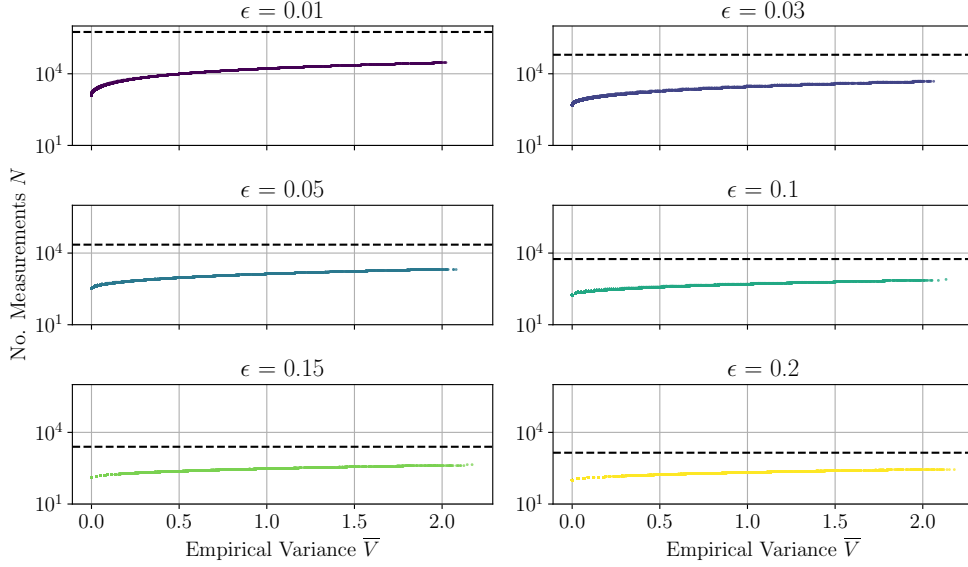


Figure 16: This figure shows the amount of samples EBS used, as a function of the estimated variance for different values of  $\epsilon$ . The dashed line corresponds to  $t_{\min}$  given by Höfdding's bound from eq. (16). Each dot is a single step across different runs of the VQE with the same parameters. EBA requires fewer samples than Höfdding's bound across all settings of  $\epsilon$  and across the whole variance spectrum. The data used in these plots are the same as in figure 15. For the EBA variation, EBAGeoMarg alg. [4] was used, as it is the best performing.

In addition, EBS is benchmarked against the sample complexity provided by Höfdding's bound  $t_{\min}$  (see eq. 16). For a set value of  $\epsilon$ ,  $\delta$  and  $R$ ,  $t_{\min}$  is constant and therefore non-adaptive to the empirical variance of the samples. We notice that EBS yields the same accuracy for fewer samples, over all  $\epsilon$  and variance regimes, as Höfdding's bound. As the variance across all regimes is always smaller than the range  $R = 4$ , we expect EBS to perform as it does. The EBSGeoMarg alg. [4] provides further improvements beyond the variance advantage as well, see 3.1.4.

For a clearer picture of how  $\epsilon$  influences the amount of samples required for a single full VQE run, one can plot the averaged amount of samples per run as a function of  $\epsilon$ . Such a plot can be seen in figure 17.

Clearly there exists a crossover point for EBA and EBAGeo with respect to the non-adaptive Höfdding's bound, but not with EBAGeoMarg. Additionally, EBAGeoMarg, being the best performing version of EBA, has a significantly decreased sample complexity than the EBS variations and Höfdding's bound. This behaviour is expected as shown in section 3.1.5. Henceforth, only EBAGeoMarg is used when using an EBS algorithm, as it is the best performing EBS algorithm [23].

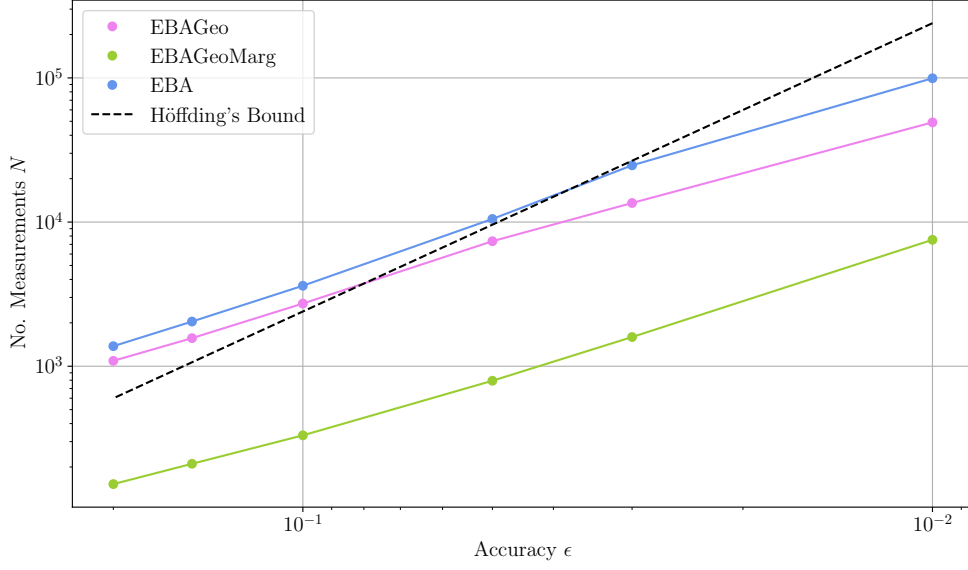


Figure 17: Sample complexity for EBS modifications and Höfding's bound. Both axes scale logarithmically. The displayed data is the mean of the number of samples, over 100 VQE runs, i.e., across all variance regimes as well. Each run is identical as the same predefined path is traversed each time. The data is the same as in figure 16. Each dot represents data of: EBA [2] (blue), by EBAGeo [3] (purple) and by EBAGeoMarg [4] (green). The dashed black line is the number of measurements given by  $t_{\min}$ , see. eq. (16). The respective coloured lines (excluding dashed lines) are added for visual guidance and do not represent any data.

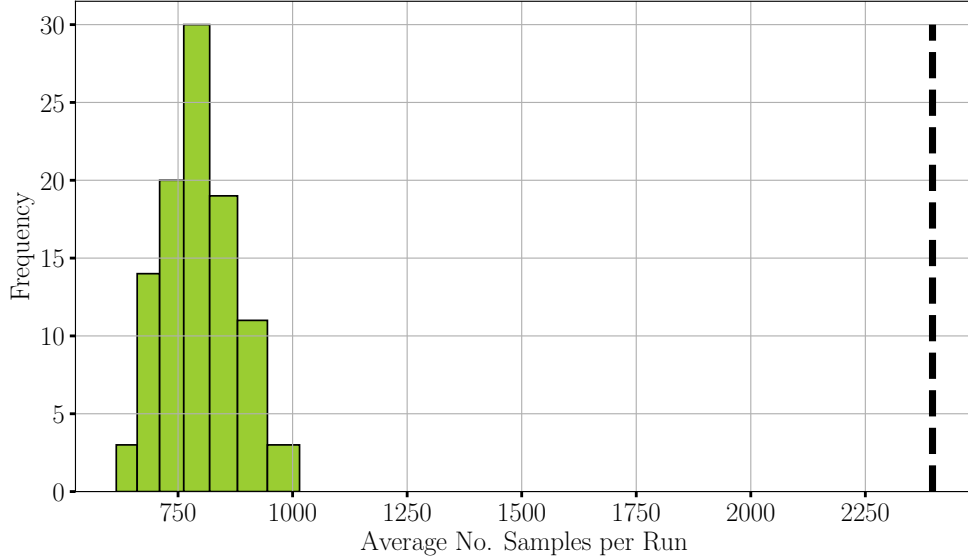


Figure 18: Histogram of the mean number of samples per VQE step. The green data is generated using the EBAGeoMarg algorithm [4], while the dashed line represents the constant number of samples given by Höfding's bound (see eq.(16)). There are 100 runs sampled using following parameters:  $\epsilon = 0.05$ ,  $\delta = 0.1$ ,  $\alpha = 0.25$  and  $\mu = 0.1$ .

EBAGeoMarg requires significantly fewer samples for a single energy measurement across all variance settings and accuracy values  $\epsilon$  than the constant sample budget given by Höfdding’s bound. Not surprisingly, figure 18 shows that EBAGeoMarg also requires less samples per run overall than Höfdding’s bound.

The variation of the stopping time, from using EBAGeoMarg, can be partially attributed to the SPSA method that was used. This method uses a randomly chosen direction for the gradient estimation, which then translates to randomness in the path taken. Most importantly, figure 18 shows the influence of the variance on EBS and Höfdding’s bound.

## 4.2. Dissociation Curve of $H_2$

While the toy example from the previous section serves as a proof of concept of EBS’ ability to reduce the number of samples required in a VQE setting, the question is now if this holds true for more practical applications. A realistic application for a VQE is, for example, solving the **electronic structure problem**.

The electronic structure problem is concerned with how the energy of a molecule depends on the specific configuration of the electrons. Before one can solve such a problem on a quantum computer, some preparation has to be performed. Firstly, the molecular Hamiltonian has to be approximated using the Born-Oppenheimer approximation, as for bigger molecules it is not analytically solvable. Secondly, the second quantisation is applied, such that the Hamiltonian is in terms of fermionic operators (creation/ annihilation operator). Lastly, using a fermion-qubit mapping results in a Hamiltonian that is in terms of Pauli operators and can be used in quantum computers. More details can be found in Ref. [27]. A flowchart of this process is shown in figure 19.

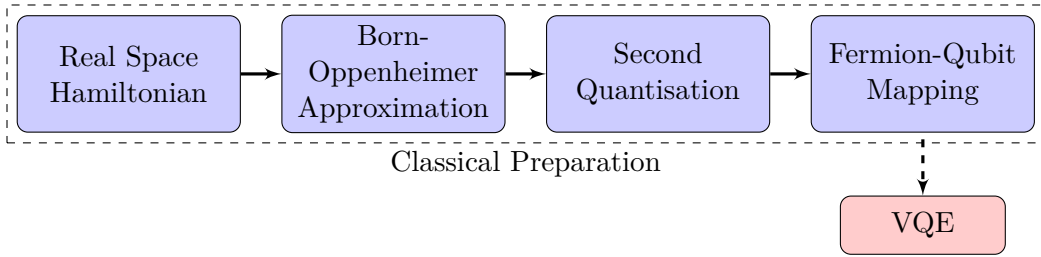


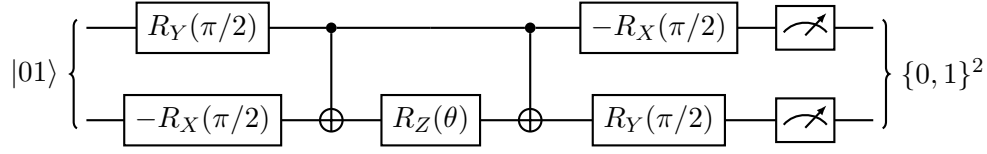
Figure 19: Flowchart for preparing an electronic structure problem for a VQE. Adapted from Ref. [27].

In this section, we look at such an electronic structure problem in the case of the  $H_2$  molecule. The energy of the  $H_2$  molecule depends on the bond length  $d$ , i.e. the distance between the two hydrogen atoms. The according Hamiltonian [27] is given as:

$$\hat{H} = g_1 \mathbb{1} + g_2 Z_0 + g_3 Z_1 + g_4 (Z_0 \otimes Z_1) + g_5 (Y_0 \otimes Y_1) + g_6 (X_0 \otimes X_1), \quad (105)$$

with  $\{g_i\}$  being a parameter that maps to a particular bond length  $d$ . These parameters are tabulated in appendix D.

The corresponding ansatz [27] for this problem is as follows:



This circuit implements the unitary  $\hat{U}(\theta) = \exp\{-i\theta X \otimes Y\}$  as it always contains the ground-state of the Hamiltonian (105) [27].

#### 4.2.1. VQE run for fixed $d$

The goal is to estimate the energy of the  $H_2$  molecule by estimating the Hamiltonian at certain bond lengths  $d$ . To this extent, an exemplary VQE run for  $d = 0.75\text{\AA}$  is shown first. The corresponding set of coefficients,  $\{g_i\}$ , can be read from appendix D.

In figure 20, a VQE run for  $d = 0.75\text{\AA}$  can be seen. The clusters of green dots, i.e. the energy estimates, appear to due to the SPSA method. Furthermore, some variance of the energy for a certain  $\theta$  is to be expected and is controlled by the estimation promise of EBA.

Here, we chose  $\epsilon = 0.01$ ,  $\delta = 0.1$ ,  $\alpha = 0.1$  and  $\mu = 0.1$ . Running this VQE for an array of bond lengths  $d$  gives the energy curve of the  $H_2$  molecule based on the bond length. However, this exemplary VQE run serves as an analysis tool to assess the quality of EBS.

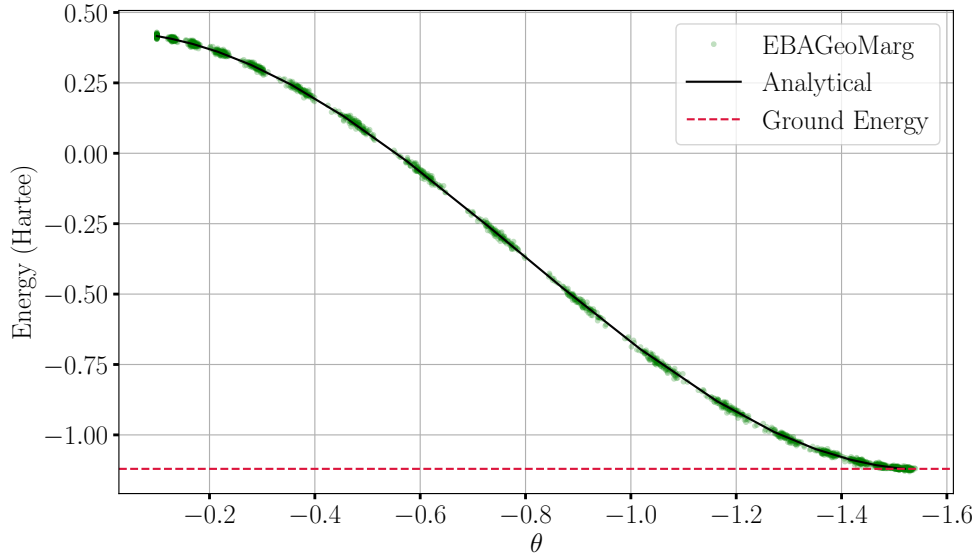


Figure 20: Typical VQE run of  $H_2$  molecule for a bond length  $d = 0.75\text{\AA}$ . The green dots are the energy estimates made with the EBAGeoMarg algorithm [4]. Additionally, the expected energy for a particular parameter  $\theta$  is displayed here as a black line, while the ground energy is represented as a red dashed line. This figure represents 100 VQE runs with an initial starting parameter of  $\theta_0 = -0.1$ . The measurement strategy, in this case, is to measure each non-commuting term individually and combine them to one estimate (see. sec. 4.2.2). Note that the  $x$ -axis is inverted, so that the VQE optimises towards the right.

#### 4.2.2. Comparing Measurement Strategies

In the section discussing the toy example (sec. 4.1), the Hamiltonians terms (eq. (103)) commute with each other, allowing simultaneous measurements of each term and, thus, a direct measurement of the energy. This is generally not the case. Already for the  $H_2$ -VQE example above, the terms of Hamiltonian (eq. (105)) do not all commute with each other. This opens up several possible measurement strategies. In the following, the goal is to find a suitable measurement strategy for this problem instance.

After inspecting the few terms in eq. (105), one suitable strategy consists of measuring once in three different Pauli bases, as indicated in table 3. In order to obtain the energy subsequently, the intermediate results have to be summed up.

Table 3: Transformations for  $\hat{H}$  (see eq. (105))

$i$ -th Term	$\hat{H}_i$	Measurement Basis
1	$\mathbb{1}$	None (offset)
2	$Z \otimes \mathbb{1}$	ZZ
3	$\mathbb{1} \otimes Z$	
4	$Z \otimes Z$	
5	$Y \otimes Y$	YY
6	$X \otimes X$	XX

Because the terms  $\hat{H}_{2,3,4}$  do commute, only one measurement is needed in the computational basis. For terms  $\hat{H}_5$  and  $\hat{H}_6$ , one measurement each in the according basis is made. This results in a total of three measurements that are needed for a single estimate. Henceforth, we refer to this method as the naive method. A summary of the terms of the  $H_2$ -Hamiltonian and their respective measurement basis, can be seen in table 3.

A measurement in the  $ZZ$  basis, here, refers to measuring the first and the second qubit in the computational basis. For measurements in the  $XX$  and  $YY$ , a unitary basis transformation, as shown in table 3 has to be applied to each measured qubit.

Another measurement strategy is the so-called L1 sampler, introduced by Arrasmith *et al.* [28]. The L1-sampler only requires one measurement per estimate, regardless of the number of Hamiltonian terms. The aforementioned, is facilitated by introducing a random sampling process that selects a Hamiltonian term, such that the term with the largest coefficient in absolute values is the one most likely to be sampled and measured. Each Hamiltonian term is drawn with the probability, as follows:

$$p_i(d) = \frac{|g_i(d)|}{\|g(d)\|_1}, \quad (106)$$

with  $\|g(d)\|_1 = \sum_{k=1}^N |g_k|$  and  $N$  being the number of terms in the Hamiltonian.

However, this sampling procedure increases the variance of the estimator, hence it is unclear if this method ultimately reduces the required number of shots for an accurate estimation.

In general, it is unclear which measurement strategy is suitable when used in conjunction with EBS. Hence, we benchmark both estimation schemes in the

optimization of an VQE for the  $H_2$ -Hamiltonian, see eq. 105. In particular, we track the empirical variances and the corresponding measurement rounds during the optimization. This effect can be seen in 21. The variance, estimated by the L1-sampler, does not converge to zero as the unbiased variance does, which is mirrored in the number of measurements needed for this method.

As can be seen in figure 21, over a whole VQE optimisation run, the naive method requires fewer samples in total than the L1-sampler. While the L1-sampler requires  $4.2 \times 10^5$  samples total, the naive method only requires  $2.9 \times 10^5$  samples, i.e., saving around a third of the measurement repetitions to provably reach the same accuracy.

While this generally might not be the case, for this particular problem, the naive method is the preferable measurement strategy as the L1-sampler significantly increased the variance across all regimes. Henceforth, the naive method is used as the measurement strategy, as it is optimal for this particular problem.

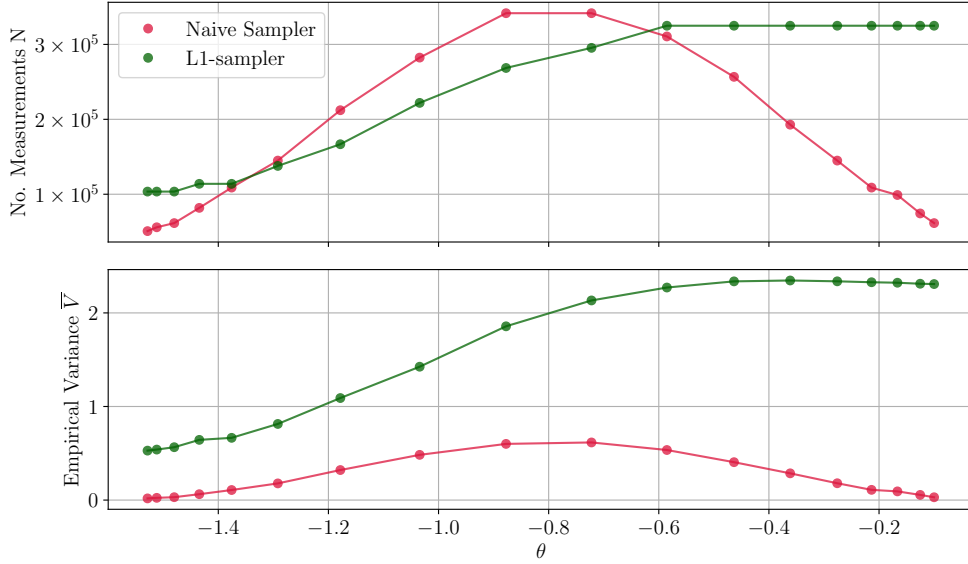


Figure 21: Empirical variance  $\bar{V}$  and the number of measurements  $N$ , as a function of the optimisation parameter  $\theta$ . The red dots (line included as visual aid) correspond to the naive method, while the green dots correspond to the L1-sampler. For the energy and variance estimation the EBAGeoMarg algorithm 4 is used with a respective measurement method. Following parameters for the VQE are used:  $\alpha = 0.1$ ,  $\mu = 0.1$ ,  $\delta = 0.1$  and  $\epsilon = 0.01$ . Additionally, the parameters  $\{g_i\}$  correspond to a bond length  $d = 0.75\text{\AA}$  and can be seen in appendix D. Note that the  $x$ -axis is inverted, such that the VQE optimises towards the right.

#### 4.2.3. Effect of $\epsilon$ on Actual Accuracy and Success Probability

Quantum chemistry problems require a certain accuracy, thus the scaling of the sampling complexity based on the parameter  $\epsilon$  is of importance. Similarly to the analysis of the toy example in section 4.1.1, the effect of the accuracy parameter  $\epsilon$  on the empirically measured accuracy and confidence is analysed. To that extent, the VQE for  $H_2$  is run for a fixed bond length  $d = 0.75\text{\AA}$  with

variable values of  $\epsilon$  as input for the EBS scheme.

Figure 22 shows that EBS returns an estimate below the required accuracy with an empirical probability of at least  $1 - \delta = 0.9$  of the times. Likewise to the toy example, a significantly higher accuracy than  $\epsilon$  is observed. This effect is more pronounced the higher the accuracy is, or the lower  $\epsilon$  is set. Further analysis on this effect is the subject of the following section.

As mentioned before, this is to be expected, as EBS in general has overhead reserved to fulfil the guarantees of an  $(\epsilon, \delta)$ -estimate. Nonetheless, the confidence parameter  $\delta = 0.1$  requires EBS to produce an  $\epsilon$ -accurate estimation with empirical probability of around 10%. Again, the confidence  $1 - \delta$  is not compared here, as one can always achieve higher confidences by repeating the experiment  $n$  times. This leads to a decrease of  $\delta$  as estimates are deluded with more  $(\epsilon, \delta)$ -estimates [14].

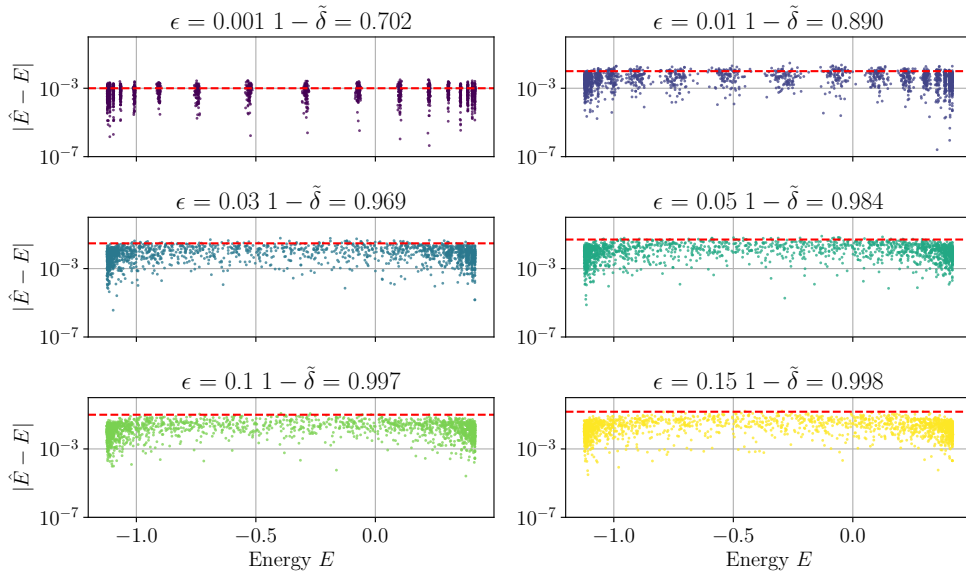


Figure 22: This figure shows the distance/ accuracy  $|\hat{E} - E|$  as a function of the analytical energy  $E$  for different values of  $\epsilon$ , with  $\hat{E}$  being the estimated energy. The red line is the corresponding value of  $\epsilon$ . Each dot is a single step of various runs of the VQE.  $1 - \tilde{\delta}$  refers to the measured confidence, i.e.,  $\mathbb{P}[|\hat{E} - E| \leq \epsilon]$ . Following parameters for the VQE are used:  $\alpha = 0.1$ ,  $\mu = 0.1$  and  $\delta = 0.1$ . In addition, we use  $\epsilon = \{0.001, 0.01, 0.03, 0.05, 0.1, 0.15\}$  and the EBAGeoMarg algorithm [4]. The actual accuracy  $|\hat{E} - E|$  is usually below the set accuracy  $\epsilon$  (red dashed line).

#### 4.2.4. Effect of $\epsilon$ on Sample Complexity

Especially in the context of quantum chemistry, the accuracy  $\epsilon$  is of importance as a certain accuracy is always required for further calculations. Again, we fix the bond length  $d = 0.75\text{\AA}$  while varying over an array of  $\epsilon$ , to analyse how EBS compares to the non-adaptive Hoeffding's bound.

In figure 23 the number of measurements, as given by EBS and  $t_{\min}$ , can be seen as a function of the empirical variance intervals. A similar behaviour as with the toy example in section 4.1 can be observed here. EBS requires more samples in higher variance regimes but always less than the constant  $t_{\min}$  pro-



vided by Höfdding’s bound over all variance regimes. Note that the highest variance in this case is  $\approx 0.6$ , while the range is  $R = 3.0636$ . Hence, following prior analysis, see sec. 2.2.4, EBS is expected to perform better.

Additionally, an artefact of the EBAGeoMarg [4] algorithm can be seen where the number of measurements is constant for some variances. This is due to the fact that these algorithms build up an overhead by virtue of the geometric sampling procedure, see sec. 3.1.3.

Similar scaling in figure 24, for higher accuracies the difference between EBS and Höfdding’s is increasing. These differences, which correspond to the measurement savings, are significant even for the lowest accuracies. In these settings, EBS manages to decrease the measurement load by up to two orders of magnitude.

Keeping in mind that in the context of actual quantum hardware, samples are a valuable resource, and minimising the amount of measurements, i.e., the samples, is always of benefit. This effect is further quantified by in figure 24. The scaling of EBS (EBAGeoMarg [4]) clearly outperforms Höfdding’s bound, i.e., the discrepancy between the two bounds only increases with higher accuracy levels  $\epsilon$ . This scaling behaviour is expected following the analysis of the respective bound in section 2.2.4 and Ref. [23].

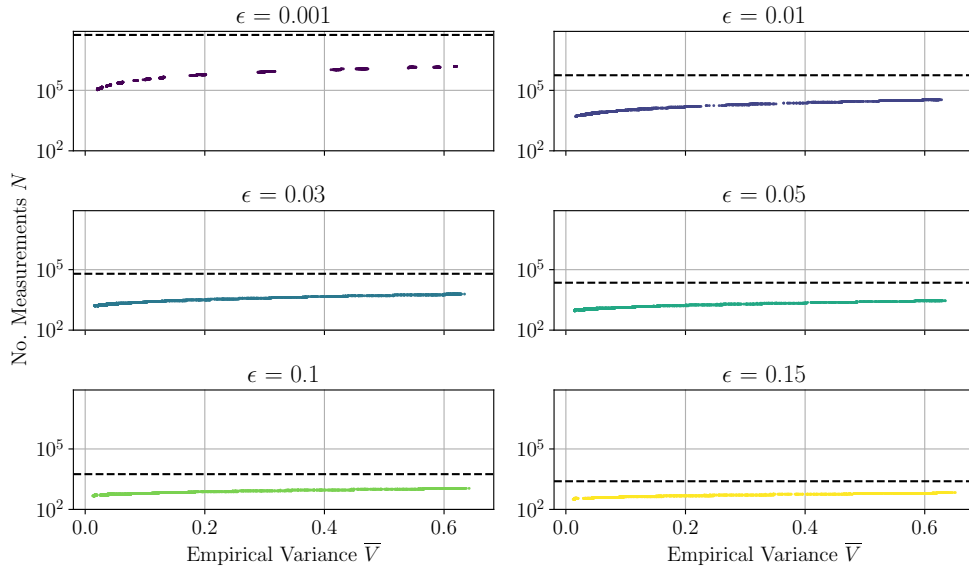


Figure 23: Amount of samples EBS uses, as a function of the estimated variance for different values of  $\epsilon$ . The dashed line corresponds to  $t_{\min}$  given by Höfdding’s bound from eq. (16). Each dot is a single step of various runs of the VQE initialised with the same parameters. EBA requires less samples than Höfdding’s bound across all settings of  $\epsilon$  and across the whole variance spectrum. The data used in these plots is the same as in figure 22. For the EBA variation EBAGeoMarg is used, as it is the best performing. For these plots the same data as in figure 22 is used.

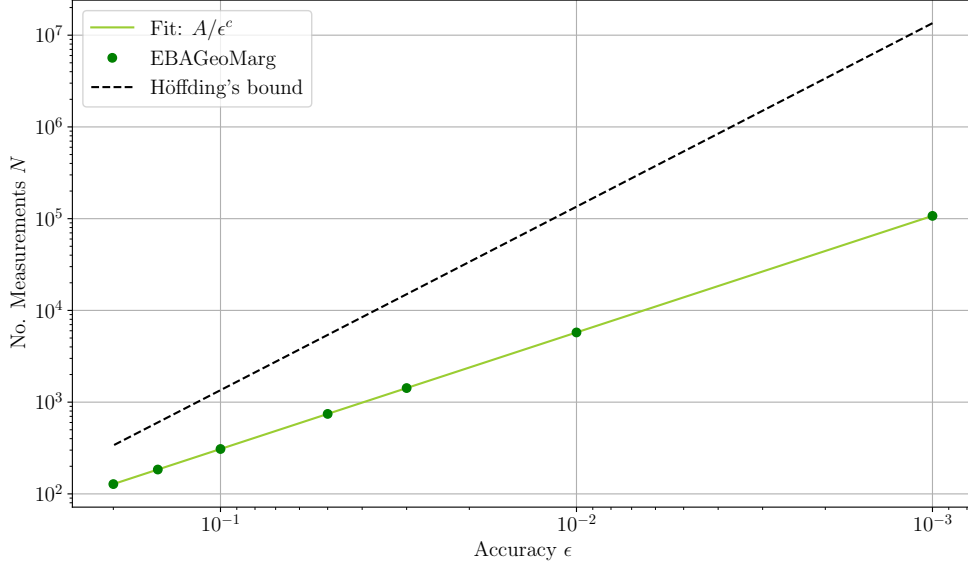


Figure 24: Number of measurements as a function of accuracy  $\epsilon$ . Here, EBAGeoMarg [4] in green (dots) and  $t_{\min}$  (16), in black (dashed line) are compared. Each data point of EBAGeoMarg, represented as a point, is averaged over 100 distinct VQE runs. Furthermore, the green line is fitted to the EBAGeoMarg data points, according to  $\frac{A}{\epsilon^c}$  with  $A = 13.83$  and  $c = 1.31$ . Furthermore, the x-axis is inverted so that this plot reads from right to left, i.e., the accuracy increases to the right. Note that EBAGeoMarg scales better with increasing accuracy, as compared to Höfding's bound, where  $c = 2$ . For this plot, the same data as in figure 22 is used.

#### 4.2.5. Dissociation Curve

Running a VQE for an array of bond lengths  $d$ , e.g. as in section 4.2.1, gives a (total) energy curve of the  $H_2$  molecule depending on the distance of the two hydrogen atoms. Each energy measurement is made using EBS (EBAGeoMarg [4]) and therefore an accuracy of  $\epsilon$  with the confidence  $1 - \delta$  is guaranteed.

Such an energy curve for  $H_2$  can be seen in figure 25. Here, the chosen circuit ansatz faithfully reproduces (after optimization of its parameter) the correct energies at all bond lengths up to an accuracy level  $\epsilon = 0.01$ .

Figure 25 also shows the scaling of EBS with the range of the energy and therefore of the samples. As the range of the random variables depends on the bond length  $d$ , or more specifically the corresponding parameters  $\{g_i\}$ , EBS requires less samples for certain bond lengths. Note that despite the fact that Höfding's bound is also depended on the range and a similar downward trend can be seen, EBS provides a better bound over all  $d$ . This is to be expected, as EBS performs better over all variance regimes (see figure 23).

Additionally, the number of measurements averaged over a VQE run decreases with  $d$ . This is due to the range also decreasing with  $d$ , as EBS is dependent on the range due to the empirical Bernstein bound (see eq. (19)). The range  $R$  for different bond lengths  $d$  is calculated as follows:

$$R(d) = 2\|g(d)\|_1 \rightarrow E(d) \in [-R(d)/2, R(d)/2]. \quad (107)$$

Additionally, the number of iterations the VQE needs is fluctuating due to constant hyperparameters for the gradient descent and a constant starting parameter. By optimising these parameters, one could achieve a reduction in measurements by virtue of shorter VQE runs. However, parameter optimisation is not the focus of this thesis.

Furthermore, the Python code for the VQE simulation can be found in the GitHub repository **EmpiricalBernsteinAlgorithm** [24] as an application for EBS.

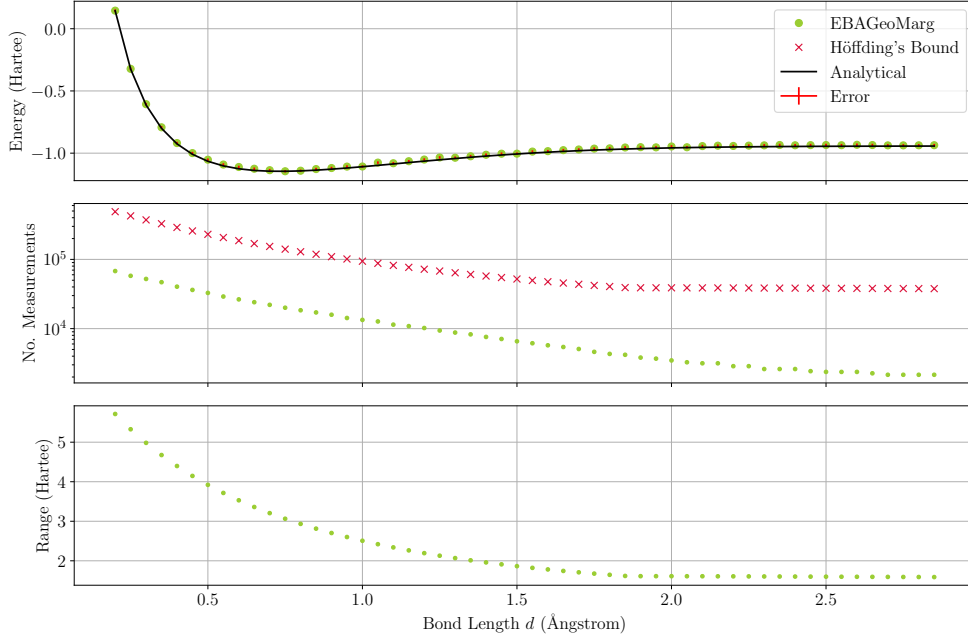


Figure 25: This plot shows the  $H_2$  energy curve as a function of the bond length  $d$ , alongside with averaged number of measurements over a VQE run and the range of the energy depending on  $d$ . The coefficients  $\{g_i\}$  are tabulated in appendix D. For the energy estimation (green dots) (EBAGeoMarg [4]) is used with accuracy  $\epsilon = 0.01$  and  $\delta = 0.1$ . The red crosses correspond to the number of measurements given by  $t_{\min}$  (see eq. 16).

## 5. Conclusion

In this thesis, we explore the use of the empirical Bernstein stopping (EBS) algorithm as a method to reduce the sample complexity in hybrid quantum algorithms. More specifically, EBS is used inside a VQE, where it estimates the desired ground state energy as well as energies at different parameters for the gradient estimation. In particular, EBS is used to obtain energy estimates while also provably providing an estimate with accuracy  $\epsilon$  with a confidence of  $1 - \delta$ .

To this extent, firstly a toy example with a simple Hamiltonian is constructed using methods that yield a quantum circuit that diagonalizes the Hamiltonian. Consequently, the resulting ansatz contains the solution to our problem, i.e. the lowest eigenvalue of the Hamiltonian, as it is analytically equivalent to the Hamiltonian. EBS then is used to estimate the energy at each VQE step as well as its gradient that is needed for the optimisation method used. In the case of the toy example, EBS reduces the sample complexity, i.e. reduces the number of measurements required for an accurate energy estimation.

Surprisingly, EBS managed to outperform the non-adaptive Höfdding’s bound, not only in low variance regimes but, somewhat surprisingly, also over a whole VQE optimisation run (see section 4.1.2). Additionally, in section 4.1.1, the measured accuracy matches the required accuracy  $\epsilon$  with confidence  $1 - \delta$ .

As EBS performs well above our expectations in the toy example (sec. 4.1), the next step involves an application of quantum chemistry. In this thesis, we chose to solve the electronic structure problem for the  $H_2$  molecule using an ansatz used by O’Malley *et al.* [27]. In contrast to our constructed ansatz for the toy example, the ansatz for the  $H_2$  molecule is a physically motivated one but nevertheless just an ansatz. This problem required a closer look at the specific measurement strategy, as the Hamiltonian contains non-commuting terms that necessitate multiple measurements for a single estimate. To this extent, the L1-sampler, proposed by [28], is compared with a naive method, as it allows an energy estimation to be made using only a single measurement. The naive method proves to be more efficient in this case as the L1-sampler increases - due to the additional sampling routine - the estimator’s variance which, in turn, increases the number of measurements needed for an energy estimation.

Analysis of the sample complexity and accuracy of the estimates (section 4.2.3 & 4.2.4), again, shows that EBS does not only uphold its guarantees but also does so by utilising fewer measurements as compared to Höfdding’s bound. A tangible comparison between EBS and Höfdding’s bound is summarised in figure 24, where the EBS clearly performs better. Using EBS for all required energy measurements, the energy curve of a hydrogen molecule, depending on the bond length  $d$ , is presented in figure 25 as the concluding result.

The conducted simulations of a hydrogen molecule and a toy example have shown the strengths of EBS and its modifications in the context of variational quantum algorithms/ eigensolvers and quantum computing in general. As measurements are an integral part of all quantum computing applications, these results are promising for other applications outside of VQAs as well.

## 6. Outlook

Both the toy and quantum chemistry example and the electronic structure problem for  $H_2$ , prove that one can utilise the adaptive stopping algorithm EBS, to reduce the measurement effort. This outperforms the conservative Hoeffding’s bound which readily serves as a benchmarking tool. While the  $H_2$  molecule provides a more complex and realistic problem, it is, nevertheless, a two-qubit problem, i.e., it can easily be solved classically as well as analytically. It is generally not clear which particular ansatz or measurement strategy is the optimal one for quantum chemistry problems. The behaviour of EBS for larger systems and more complicated Hamiltonians is still unclear, as the measurement strategy heavily influences the energy or variance estimator. Moreover, the results of these simple problems suggest similar properties for more involved problems.

Additionally, the simulation of the quantum computer for these experiments did not include noise simulation. It is again unclear how noise influences the energy and variance estimator and how large the measurement overhead then is. More specifically, it is unclear if EBS will keep its advantage over non-adaptive methods such as Hoeffding’s bound.

Secondly, (empirical) Bernstein inequality, bound and algorithm are defined for real-valued random variables, which limits its applicability for methods that deal with other random variables. Adapting EBS to random variables beyond real valued ones, i.e. random vectors or matrices, makes EBS applicable to more sophisticated measurement methods of quantum many-body Hamiltonians [29].

## A. Bernstein Bound Derivation

Let  $X_1, \dots, X_t$  be i.i.d random variables with range  $R$ , expected value  $\mu$  and variance  $\sigma$ . Let  $\bar{X}_t = \frac{1}{t} \sum_{i=1}^t X_i$  and  $\Sigma^2 = \sum_{i=1}^t \sigma_i^2 \stackrel{\text{i.i.d.}}{=} t\sigma^2$ .

$$\mathbb{P}[|\bar{X}_t - \mu| \geq \epsilon] \leq 2 \exp \left( -\frac{(t\epsilon)^2/2}{\Sigma^2 + Rt\epsilon/3} \right) \stackrel{!}{=} \delta \quad (\text{A.1})$$

The equation above can be rearranged in form more suitable for algorithms.

$$|\bar{X}_t - \mu| \geq \epsilon \quad (\text{A.2})$$

$\epsilon$  can be calculated using the right-hand side of equation (A.1).

$$\delta = 2 \exp \left( -\frac{(t\epsilon)^2/2}{t\sigma^2 + Rt\epsilon/3} \right) \quad (\text{A.3})$$

Taking the natural log on both side and rearranging gives us:

$$\begin{aligned} \epsilon^2 &= 2 \underbrace{\ln(2/\delta)}_{:=\alpha} \left( \frac{\sigma^2}{t} + \frac{R\epsilon}{3t} \right) \\ 0 &= \epsilon^2 - \frac{2R\alpha}{3t}\epsilon - \frac{2\alpha\sigma^2}{t} \end{aligned}$$

Using the binomial theorem on the equation above gives us:

$$\left( \epsilon - \frac{R\alpha}{3n} \right)^2 = \left( \frac{R\alpha}{3n} \right)^2 + \frac{2\alpha\sigma^2}{t} \quad (\text{A.4})$$

Solving for  $\epsilon$ :

$$\begin{aligned} \epsilon &= \frac{R\alpha}{3t} \pm \sqrt{\left( \frac{R\alpha}{3t} \right)^2 + \frac{2\alpha\sigma^2}{t}} \stackrel{!}{\geq} 0 \\ \epsilon &= \frac{R\alpha}{3t} + \sqrt{\left( \frac{R\alpha}{3t} \right)^2 + \frac{2\alpha\sigma^2}{t}} \end{aligned} \quad (\text{A.5})$$

$$\geq \sigma \sqrt{\frac{2 \ln 2/\delta}{t}} + \frac{R \ln 2/\delta}{3t} \quad (\text{A.6})$$

Inserting the equation above into equation (A.2):

$$|\bar{X}_t - \mu| \geq \sigma \sqrt{\frac{2 \ln(2/\delta)}{t}} + \frac{R \ln(2/\delta)}{3t} \quad (\text{A.7})$$

While this looks similar to the empirical Bernstein bound (19), this equation uses the standard deviation  $\sigma$  not the empirical standard deviation  $\bar{\sigma}_t$ .

## B. Quantum Mechanics/ Computing

### B.1. Exponential Operator Derivation

Let  $\hat{A}_{n \times n}$ <sup>7</sup> be an operator that fulfils  $\hat{A}^2 = \mathbb{1}_{n \times n}$  and  $\theta \in \mathbb{R}$  then the following equation holds:

$$e^{i\theta\hat{A}} = \cos(\theta)\mathbb{1} + i \sin(\theta)\hat{A}. \quad (\text{B.1})$$

This can be shown by expanding the exponential function with the Taylor series:

$$\begin{aligned} e^{i\theta\hat{A}} &= \sum_{k=0}^{\infty} \frac{(i\theta\hat{A})^k}{k!} \\ &= \hat{A}^0 + i\theta\hat{A} + \frac{(i\theta\hat{A})^2}{2!} + \frac{(i\theta\hat{A})^3}{3!} + \dots \\ &= \mathbb{1} + i\theta\hat{A} - \frac{(\theta\hat{A})^2}{2!} - i\frac{(\theta\hat{A})^3}{3!} + \frac{(\theta\hat{A})^4}{4!} + \dots \\ &= \underbrace{\left(1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} + \dots\right)}_{\cos(\theta)} \mathbb{1} + i \underbrace{\left(\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} + \dots\right)}_{\sin(\theta)} \hat{A} \\ &= \cos(\theta)\mathbb{1} + i \sin(\theta)\hat{A}. \end{aligned} \quad (\text{B.2})$$

### B.2. Parameter Shift Rule

A method to calculate the gradient of a function is the so-called **Parameter shift rule**. It makes use of the fact that functions like equation (72) can be written in terms of sin and cos terms [30]. This makes it possible to write the gradient  $\nabla C(\boldsymbol{\theta}_k)$ , for the  $k$ -th parameter, of a function  $C$  in terms of itself shifted in its arguments.

$$\nabla C(\boldsymbol{\theta}_k) = \frac{C(\boldsymbol{\theta}_k + \boldsymbol{\mu}_k) - C(\boldsymbol{\theta}_k - \boldsymbol{\mu}_k)}{2 \sin \mu}, \quad (\text{B.3})$$

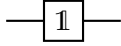
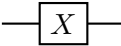
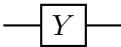
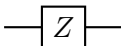
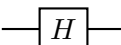
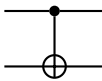
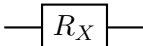
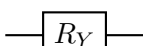
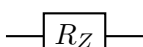
with  $\mu \neq n\pi$  and  $\boldsymbol{\mu}_k = \mu \hat{e}_k$ .

---

<sup>7</sup>Any square matrix  $M_{n \times n}$  also fulfils this equation:  $M_{n \times n}^0 = \mathbb{1}_{n \times n}$

### B.3. Quantum Gates

Table 4: Summary of the gates introduced in this section.

Gate	Circuit symbol	Matrix
$\mathbb{1}$		$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
$X$		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
$Y$		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
$Z$		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
$H$		$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
CNOT		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
$R_X(\theta)$		$\begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -i \sin\left(\frac{\theta}{2}\right) \\ -i \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$
$R_Y(\theta)$		$\begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$
$R_Z(\theta)$		$\begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$



## C. Quantum Circuit Decomposition: Example Circuit

Let  $\hat{H} = (Z \otimes \mathbb{1}) + (\mathbb{1} \otimes X)$ . As a matrix this can be written as:

$$\hat{H} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \quad (\text{C.1})$$

The corresponding eigenstates are:

$$|\psi_1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, |\psi_2\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, |\psi_3\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, |\psi_4\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \end{bmatrix}, \quad (\text{C.2})$$

with  $\lambda_1 = 2$ ,  $\lambda_2 = \lambda_3 = 0$  and  $\lambda_4 = -2$ .

Knowing the the eigenstates and the eigenvalues, the eigenvalue decomposition of  $\hat{H}$  is as follows:

$$\begin{aligned} \hat{H} &= \hat{U} \Lambda \hat{U}^\dagger \\ &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix}. \end{aligned} \quad (\text{C.3})$$

## D. Coefficients $\{g_i\}$ for eq. (105)

The coefficients are readily provided in Ref. [27] and listed here again for convenience.

$d$	$g_1$	$g_2$	$g_3$	$g_4$	$g_5$	$g_6$
0.20	2.8489	0.5678	-1.4508	0.6799	0.0791	0.0791
0.25	2.1868	0.5449	-1.2870	0.6719	0.0798	0.0798
0.30	1.7252	0.5215	-1.1458	0.6631	0.0806	0.0806
0.35	1.3827	0.4982	-1.0226	0.6537	0.0815	0.0815
0.40	1.1182	0.4754	-0.9145	0.6438	0.0825	0.0825
0.45	0.9083	0.4534	-0.8194	0.6336	0.0835	0.0835
0.50	0.7381	0.4325	-0.7355	0.6233	0.0846	0.0846
0.55	0.5979	0.4125	-0.6612	0.6129	0.0858	0.0858
0.60	0.4808	0.3937	-0.5950	0.6025	0.0870	0.0870
0.65	0.3819	0.3760	-0.5358	0.5921	0.0883	0.0883
0.70	0.2976	0.3593	-0.4826	0.5818	0.0896	0.0896
0.75	0.2252	0.3435	-0.4347	0.5716	0.0910	0.0910
0.80	0.1626	0.3288	-0.3915	0.5616	0.0925	0.0925
0.85	0.1083	0.3149	-0.3523	0.5518	0.0939	0.0939
0.90	0.0609	0.3018	-0.3168	0.5421	0.0954	0.0954
0.95	0.0193	0.2895	-0.2845	0.5327	0.0970	0.0970
1.00	-0.0172	0.2779	-0.2550	0.5235	0.0986	0.0986
1.05	-0.0493	0.2669	-0.2282	0.5146	0.1002	0.1002
1.10	-0.0778	0.2565	-0.2036	0.5059	0.1018	0.1018
1.15	-0.1029	0.2467	-0.1810	0.4974	0.1034	0.1034

$d$	$g_1$	$g_2$	$g_3$	$g_4$	$g_5$	$g_6$
1.20	-0.1253	0.2374	-0.1603	0.4892	0.1050	0.1050
1.25	-0.1452	0.2286	-0.1413	0.4812	0.1067	0.1067
1.30	-0.1629	0.2203	-0.1238	0.4735	0.1083	0.1083
1.35	-0.1786	0.2123	-0.1077	0.4660	0.1100	0.1100
1.40	-0.1927	0.2048	-0.0929	0.4588	0.1116	0.1116
1.45	-0.2053	0.1976	-0.0792	0.4518	0.1133	0.1133
1.50	-0.2165	0.1908	-0.0666	0.4451	0.1149	0.1149
1.55	-0.2265	0.1843	-0.0549	0.4386	0.1165	0.1165
1.60	-0.2355	0.1782	-0.0442	0.4323	0.1181	0.1181
1.65	-0.2436	0.1723	-0.0342	0.4262	0.1196	0.1196
1.70	-0.2508	0.1667	-0.0251	0.4204	0.1211	0.1211
1.75	-0.2573	0.1615	-0.0166	0.4148	0.1226	0.1226
1.80	-0.2632	0.1565	-0.0088	0.4094	0.1241	0.1241
1.85	-0.2684	0.1517	-0.0015	0.4042	0.1256	0.1256
1.90	-0.2731	0.1472	0.0052	0.3992	0.1270	0.1270
1.95	-0.2774	0.1430	0.0114	0.3944	0.1284	0.1284
2.00	-0.2812	0.1390	0.0171	0.3898	0.1297	0.1297
2.05	-0.2847	0.1352	0.0223	0.3853	0.1310	0.1310
2.10	-0.2879	0.1316	0.0272	0.3811	0.1323	0.1323
2.15	-0.2908	0.1282	0.0317	0.3769	0.1335	0.1335
2.20	-0.2934	0.1251	0.0359	0.3730	0.1347	0.1347
2.25	-0.2958	0.1221	0.0397	0.3692	0.1359	0.1359
2.30	-0.2980	0.1193	0.0432	0.3655	0.1370	0.1370
2.35	-0.3000	0.1167	0.0465	0.3620	0.1381	0.1381
2.40	-0.3018	0.1142	0.0495	0.3586	0.1392	0.1392
2.45	-0.3035	0.1119	0.0523	0.3553	0.1402	0.1402
2.50	-0.3051	0.1098	0.0549	0.3521	0.1412	0.1412
2.55	-0.3066	0.1078	0.0572	0.3491	0.1422	0.1422
2.60	-0.3079	0.1059	0.0594	0.3461	0.1432	0.1432
2.65	-0.3092	0.1042	0.0614	0.3433	0.1441	0.1441
2.70	-0.3104	0.1026	0.0632	0.3406	0.1450	0.1450
2.75	-0.3115	0.1011	0.0649	0.3379	0.1458	0.1458
2.80	-0.3125	0.0997	0.0665	0.3354	0.1467	0.1467
2.85	-0.3135	0.0984	0.0679	0.3329	0.1475	0.1475

## References

- [1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, *et al.*, *Quantum supremacy using a programmable superconducting processor*, *Nature* **574**, 505 (2019).
- [2] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, *et al.*, *Quantum computational advantage using photons*, *Science* **370**, 1460 (2020).
- [3] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, *et al.*, *Variational quantum algorithms*, *Nature Reviews Physics* **3**, 625 (2021).
- [4] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, *Barren plateaus in quantum neural network training landscapes*, *Nature communications* **9**, 4812 (2018).
- [5] L. Bittel and M. Kliesch, *Training variational quantum algorithms is np-hard*, *Physical review letters* **127**, 120502 (2021).
- [6] L. Bittel, J. Watty, and M. Kliesch, *Fast gradient estimation for variational quantum algorithms*, [arXiv:2210.06484 \[quant-ph\]](https://arxiv.org/abs/2210.06484) (2022).
- [7] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O’Brien, *A variational eigenvalue solver on a photonic quantum processor*, *Nature Communications* **5**, 4213 (2014).
- [8] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, *Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices*, *Physical Review X* **10**, 021067 (2020).
- [9] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, *An adaptive variational algorithm for exact molecular simulations on a quantum computer*, *Nature communications* **10**, 3007 (2019).
- [10] H. R. Grimsley, G. S. Barron, E. Barnes, S. E. Economou, and N. J. Mayhall, *Adaptive, problem-tailored variational quantum eigensolver mitigates rough parameter landscapes and barren plateaus*, *npj Quantum Information* **9** (2023), [10.1038/s41534-023-00681-0](https://doi.org/10.1038/s41534-023-00681-0).
- [11] J. K. Blitzstein and J. Hwang, *Introduction to probability* (Crc Press Boca Raton, FL, 2015).
- [12] T. Popoviciu, *Sur les équations algébriques ayant toutes leurs racines réelles*, *Mathematica* **9**, 20 (1935).
- [13] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis* (Cambridge university press, 2017).
- [14] M. Kliesch, *Characterization, certification, and validation of quantum systems*, (2020).

- [15] V. Mnih, C. Szepesvári, and J.-Y. Audibert, *Empirical bernstein stopping*, *Proceedings of the 25th international conference on Machine learning*, , 672 (2008).
- [16] J.-Y. Audibert, R. Munos, and C. Szepesvári, *Variance estimates and exploration function in multi-armed bandit*, *CERTIS Research Report 07–31*, (2007).
- [17] J. Preskill, *Quantum computing in the NISQ era and beyond*, *Quantum* **2**, 79 (2018).
- [18] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, Vol. 54 (2010).
- [19] Smite-Meister, *Bloch sphere, a geometrical representation of a two-level quantum system*, (2009).
- [20] A. E. D. Deutsch, A. Barenco, *Universality in quantum computation*, *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* **449**, 669 (1995).
- [21] G. E. Crooks, *Gates, states, and circuits* (2020).
- [22] T. Hoffmann and D. Brown, *Gradient estimation with constant scaling for hybrid quantum machine learning*, [arXiv:2211.13981 \[quant-ph\]](#) (2022).
- [23] V. Mnih, *Efficient stopping rules* (2008).
- [24] U. Tepe, *Empiricalbernsteinalgorithm*, (2023).
- [25] V. Fedoriaka, *Decomposition of unitary matrix into quantum gates*, (2019).
- [26] V. Fedoriaka, *Tool for decomposing unitary matrix into quantum gates*, (2020).
- [27] P. J. O’Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, *et al.*, *Scalable quantum simulation of molecular energies*, *Physical Review X* **6**, 031007 (2016).
- [28] A. Arrasmith, L. Cincio, R. D. Somma, and P. J. Coles, *Operator sampling for shot-frugal optimization in variational algorithms*, [arXiv:2004.06252 \[quant-ph\]](#) (2020).
- [29] A. Gresch and M. Kliesch, *Guaranteed efficient energy estimation of quantum many-body hamiltonians using shadowgrouping*, [arXiv:2301.03385 \[quant-ph\]](#) (2023).
- [30] A. Mari, T. R. Bromley, and N. Killoran, *Estimating the gradient and higher-order derivatives on quantum hardware*, *Phys. Rev. A* **103**, 012405 (2021).