

Quantum computational finance: Monte Carlo pricing of financial derivatives

Patrick Reberstrost,^{1,*} Brajesh Gupta,^{1,†} and Thomas R. Bromley^{1,‡}

¹*Xanadu, 372 Richmond St W, Toronto, M5V 2L7, Canada*

(Dated: August 23, 2018)

This work presents a quantum algorithm for the Monte Carlo pricing of financial derivatives. We show how the relevant probability distributions can be prepared in quantum superposition, the payoff functions can be implemented via quantum circuits, and the price of financial derivatives can be extracted via quantum measurements. We show how the amplitude estimation algorithm can be applied to achieve a quadratic quantum speedup in the number of steps required to obtain an estimate for the price with high confidence. This work provides a starting point for further research at the interface of quantum computing and finance.

I. INTRODUCTION

A great amount of computational resources are employed by participants in today's financial markets. Some of these resources are spent on the pricing and risk management of financial assets and their derivatives. Financial assets include the usual stocks, bonds, and commodities, based upon which more complex contracts such as financial derivatives [1] are constructed. Financial derivatives are contracts that have a future payoff dependent upon the future price or the price trajectory of one or more underlying benchmark assets. For these derivatives, due to the stochastic nature of underlying assets, an important issue is the assignment of a fair price based on available information from the markets, what in short can be called the *pricing* problem [2, 3]. The famous Black-Scholes-Merton (BSM) model [4, 5] can price a variety of financial derivatives via a simple and analytically solvable model that uses a small number of input parameters. A large amount of research has been devoted to extending the BSM model to include complicated payoff functions and complex models for the underlying stochastic asset dynamics.

Monte Carlo methods have a long history in the sciences. Some of the earliest known applications were by Ulam, von Neumann, Teller, Metropolis *et al.* [6] in the context of the Los Alamos project, which used early computational devices such as the ENIAC. For the pricing problem in finance, the main challenge is to compute an expectation value of a function of one or more underlying stochastic financial assets. For models beyond BSM, such pricing is often performed via Monte Carlo evaluation [7].

Quantum computing promises algorithmic speedups for a variety of tasks, such as factoring or optimization. One of the earliest proposed algorithms, known as Grover's search [8], developed in the mid 1990s, in principle allows for a quadratic speed-up of searching an unstructured database. To find the solution in a size N

database with high probability, a classical computer takes $\mathcal{O}(N)$ computational steps, while a quantum computer takes $\mathcal{O}(\sqrt{N})$ steps. This algorithm has been extended and generalized to function optimization [9], amplitude amplification and estimation [10], integration [11], quantum walk-based methods for element distinctness [12], and Markov chain algorithms [13, 14], for example. In particular, the amplitude estimation algorithm can provide close to quadratic speedups for estimating expectation values [15–21], and thus provides a speedup to a problem for which Monte Carlo methods are used classically [15, 22].

Understanding the applications and enhancements of quantum mechanics to computational finance is still in its relative infancy. The framework of quantum field theory can be harnessed to study the evolution of derivatives [23]. More recent works focus on the application of quantum machine learning [24, 25] and quantum annealing [26] to areas such as portfolio optimization [27] and currency arbitrage [28]. This work investigates a new perspective of how to use quantum computing for the pricing problem. We combine well-known quantum techniques, such as amplitude estimation [10] and the quantum algorithm for Monte Carlo [15, 22] with the pricing of financial derivatives. We first show how to obtain the expectation value of a financial derivative as the output of a quantum algorithm. To this end, we show the ingredients required to set up the financial problem on a quantum computer: the elementary arithmetic operations to compute payoff functions, the preparation of the model probability distributions used in finance, and the ingredients for estimating the expectation value through an imprinted phase on ancilla qubits. It is shown how to obtain the quadratic speedup via the amplitude estimation algorithm. We discuss the quantum resources required to price European and Asian call options, representing fundamental types of derivatives. We provide evidence using classical numerical calculations that a quadratic speedup in pricing can be attained.

This article begins with a brief summary of the basics of derivative pricing. The Black-Scholes-Merton framework is introduced in Section II and classical Monte Carlo estimation is discussed in Section III within the

* pr@patrickre.com

† brajesh@xanadu.ai

‡ tom@xanadu.ai

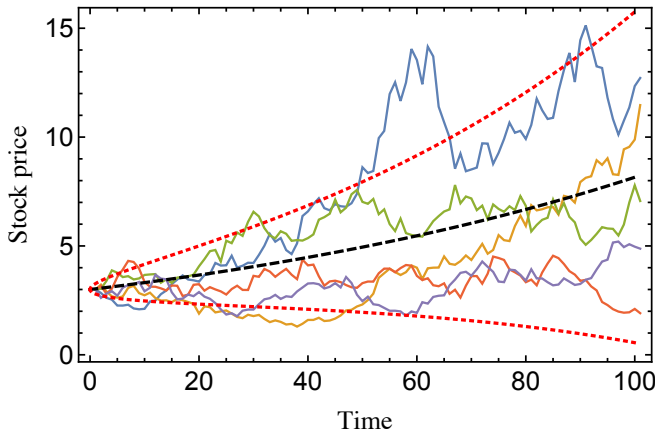


FIG. 1. The price of a stock in the Black-Scholes-Merton model behaves stochastically as a geometric Brownian motion, changing each time step according to a log-normal distribution. Here, five sample price evolutions (in dollars) of a single stock are plotted as a function of time (in days). The resultant distribution is log-normally distributed, with mean (dashed black line) and one standard deviation (dotted red lines) illustrated. Pricing an option requires estimating the expected value of a payoff function on the stock at various times. The parameters are: initial price $S_0 = \$3$, drift $\alpha = 0.1$, and volatility $\sigma = 0.25$.

context of finance. In Section IV, the quantum algorithm for Monte Carlo is given. Section V specializes this quantum algorithm to the pricing of a European call option. Section VI discusses the pricing of Asian options.

II. BLACK-SCHOLES-MERTON OPTION PRICING

The Black-Scholes-Merton (BSM) model [4, 5] considers the pricing of financial derivatives ('options'). The original model assumes a single benchmark asset ('stock'), the price of which is stochastically driven by a Brownian motion, see Fig 1. In addition, it assumes a risk-free investment into a bank account ('bond'). We follow closely the discussion in the literature [2] in the following.

Definition 1. A Brownian motion W_t is a stochastic process characterized by following attributes:

1. $W_0 = 0$;
2. W_t is continuous;
3. W_t has independent increments;
4. $W_t - W_s \sim N(0, t - s)$ for $t > s$.

Here, $N(\mu, \sigma^2)$ is the normal distribution with mean μ and standard deviation σ . Point 3 means that the random variable $W_t - W_s$ for $t > s$ is independent of any previous time random variable W_u , $u < s$. The probability

measure under which W_t is a Brownian motion shall be denoted by \mathbb{P} .

The next step is to introduce a model for the market.

Definition 2 (Black-Scholes-Merton model). The Black-Scholes-Merton model consists of two assets, one risky (the stock), the other one risk-free (the bond). The risky asset is defined by the stochastic differential equation for the price dynamics given by

$$dS_t = S_t \alpha dt + S_t \sigma dW_t, \quad (1)$$

where α is the drift, σ the volatility and dW_t is a Brownian increment. The initial condition is S_0 . In addition, the risk-free asset dynamics is given by

$$dB_t = B_t r dt, \quad (2)$$

where r is the risk-free rate (market rate). Set $B_0 = 1$. This model assumes that all parameters are constant, both assets can be bought or sold continuously and in unlimited and fractional quantities without transaction costs. Short selling is allowed, and the stock pays no dividends.

Using Ito's lemma and the fact that dW_t contributes an additional term in first order (due to its quadratic variation being proportional to dt), the risky asset stochastic differential equation can be solved as

$$S_t = S_0 e^{\sigma W_t + (\alpha - \sigma^2/2)t}, \quad (3)$$

see Appendix A. Figure 1 shows sample evolutions of S_t . The risk-free asset is solved easily as

$$B_t = e^{rt}. \quad (4)$$

This risk-free asset also is used for 'discounting', i.e. determining the present value of a future amount of money. Let the task be to price an option. One of the simplest options is the European call option. The European call option gives the owner of the option the right to buy the stock at time $T \geq 0$ for a pre-agreed price K .

Definition 3 (European call option). The European call option payoff is defined as

$$f(S_T) = \max\{0, S_T - K\}, \quad (5)$$

where K is the strike price and T the maturity date.

The task of pricing is to evaluate at present time $t = 0$ the expectation value of the option $f(S_T)$ on the stock on the maturity date. The major tenet of risk-neutral derivative pricing is that the pricing is performed under a probability measure that shall not allow for arbitrage [3]. Simply put, arbitrage is a portfolio that has, at present, an expected future value that is greater than the current price of that portfolio. In the Black-Scholes-Merton framework, the stock price has a drift α under the \mathbb{P}

measure. Any $\alpha \neq r$ allows for arbitrage under the measure \mathbb{P} . When $\alpha > r$, one can make a profit above the market rate r by investing in the stock, and when $\alpha < r$ one can make a profit by short selling the stock. Pricing of derivatives is performed under a probability measure where the drift of the stock price is exactly the market rate r . This pricing measure is denoted by \mathbb{Q} in contrast to the original measure \mathbb{P} .

More formally, the probability measure \mathbb{Q} is defined such that the discounted asset price is a martingale, i.e. the discounted expected value of the future stock price is the present day stock price itself. The martingale property is given in this context by

$$S_0 = e^{-rT} \mathbb{E}_{\mathbb{Q}}[S_T]. \quad (6)$$

Here, e^{-rT} is the discount factor, which determines the present value of the payoff at a future time, given the model assumption of a risk-free asset growing with r . In addition, $\mathbb{E}_{\mathbb{Q}}[\cdot]$ denotes the $t = 0$ expectation value under the measure \mathbb{Q} . Under this measure, investing in the stock does not, on average, return money above or below the market rate r , i.e. does not allow for arbitrage. This feature is reflected in the martingale price dynamics, which is given by

$$dS_t = S_t r dt + S_t \sigma d\tilde{W}_t, \quad (7)$$

with the solution

$$S_t = S_0 e^{\sigma \tilde{W}_t + (r - \sigma^2/2)t}. \quad (8)$$

Here, \tilde{W}_t is a Brownian motion according to Definition 1 under the martingale measure \mathbb{Q} . The martingale property of S_t is shown in Appendix A, Lemma 2.

The pricing problem is thus given by evaluating the risk-neutral price

$$\Pi = e^{-rT} \mathbb{E}_{\mathbb{Q}}[f(S_T)], \quad (9)$$

which is the quantity of interest in this paper. For the simple European call option and several other options one can analytically solve the Black-Scholes-Merton model. A proof is sketched in Appendix A.

Result 1 (Black-Scholes-Merton price). *The risk-neutral price of the call option in Eq. (5) is given by*

$$\Pi = \Phi(d_1) S_0 - \Phi(d_2) K e^{-rT}, \quad (10)$$

with

$$d_1 = \frac{1}{\sigma \sqrt{T}} \left[\log \left(\frac{S_0}{K} \right) + \left(r + \frac{\sigma^2}{2} \right) T \right], \quad (11)$$

$$d_2 = d_1 - \sigma \sqrt{T}, \quad (12)$$

and the cumulative distribution function of the normal distribution $p(x)$,

$$\Phi(x) = \int_{-\infty}^x dy p(y) := \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x dy e^{-\frac{y^2}{2}}. \quad (13)$$

In the case of complex payoff functions and/or complex asset price dynamics, options prices cannot be solved analytically and one often resorts to Monte Carlo evaluation. Nevertheless, analytical solutions as above can be used for benchmarking Monte Carlo simulations. Finally, note that there is a dynamical equation of motion for the options price Π_t in the interval $0 \leq t \leq T$, which is given by a partial differential equation. Such an equation is used in practice for ‘hedging’, i.e. safeguarding during the time $0 \leq t \leq T$ while the option is active against eventual payouts. In this work we do not consider such a differential equation but focus on the present-day options price $\Pi \equiv \Pi_0$.

III. CLASSICAL MONTE CARLO PRICING

We first provide a brief overview of Monte Carlo derivative pricing. Options are usually nonlinear functions applied to the outcomes of one or multiple underlying assets. The option payoff depends on the asset prices at specific time instances or is based on the paths of the asset prices. As discussed in the previous section, **European options depend on the asset price at a single future time.** If the nonlinear function is piecewise linear, the option can be priced analytically, similar to Result 1 and Appendix A. If there are multiple independent Brownian processes underlying the dynamics, the price can often also be determined analytically. **The need for Monte Carlo arises if the payoff function is nonlinear beyond piecewise linear or, for example, in cases when different asset prices are assumed to be correlated.** Another class of options, called American options, allow the buyer to exercise the option at any point in time between the option start and the option maturity. Such options are related to the optimal stopping problem [3] and are also priced using Monte Carlo methods [29]. Asian options depend on the average asset price during a time intervals and, if the averaging is arithmetic, may also require Monte Carlo [30].

The underlying asset prices are modeled via stochastic differential equations. **Often stock prices are taken to be log-normal stochastic processes, i.e., driven by an exponentiated Brownian motion.** In this case, when the parameters are constant, the stochastic differential equation is exactly solvable. In other cases, such as when the parameters of the model such as the volatility itself follow a stochastic differential equation, the asset price dynamics is usually not analytically solvable. The price is then determined by sampling paths of the asset dynamics. Moreover, Brownian motions are fundamentally continuous and Gaussian with exponentially suppressed tails, features which are rarely observed in real markets. Further research has considered ‘fat-tailed’ stochastic processes and Levy jump processes [31], which often also require Monte Carlo sampling.

Monte Carlo pricing of financial derivatives proceeds in the following way. Assume that the risk-neutral

probability distribution is known, or can be obtained from calibrating to market variables. Sample from this risk-neutral probability distribution a market outcome, compute the asset prices given that market outcome, then compute the option payoff given the asset prices. Averaging the payoff over multiple samples obtains an approximation of the derivative price. Assume a European option on a single benchmark asset and let the true option price be Π and $\hat{\Pi}$ be the approximation obtained from k samples. Assume that the random variable of the payoff $f(S_T)$ is bounded in variance, i.e. $\mathbb{V}[f(S_T)] \leq \lambda^2$. Then the probability that the price estimation $\hat{\Pi}$ is ϵ away from the true price is determined by Chebyshev's inequality [22]

$$\mathbb{P}[|\hat{\Pi} - \Pi| \geq \epsilon] \leq \frac{\lambda^2}{k\epsilon^2}. \quad (14)$$

For a constant success probability, we thus require

$$k = \mathcal{O}\left(\frac{\lambda^2}{\epsilon^2}\right) \quad (15)$$

samples to estimate to additive error ϵ . The task of the quantum algorithm will be to improve the ϵ dependence from ϵ^2 to ϵ , hence providing a quadratic speedup for a given error.

Before we discuss the quantum algorithm for derivative pricing, we show how to encode expectation values into a quantum algorithm and how to obtain the same ϵ dependency as the classical algorithm. We then discuss the quadratic speedup by using the fundamental quantum algorithm of amplitude estimation.

IV. QUANTUM ALGORITHM FOR MONTE CARLO

We first discuss generically the quantum algorithms to measure an expectation value and to obtain a quadratic improvement in the number of measurements [10, 15, 22]. See Appendix B for a brief introduction to the necessary elements of quantum mechanics. In the following sections, we then specialize to European and Asian options. Assume we are given an algorithm \mathcal{A} on n qubits (the subsequent discussion can also be generalized to measuring only a subset of qubits [22]). When measuring the n qubits, the algorithm produces the n -bit string result x with probability $|a_x|^2$. In addition let $v(x)$ be a function $v(x) : \{0, 1\}^n \rightarrow \mathbb{R}$ mapping from n -bit strings to reals. Here, $v(\mathcal{A})$ denotes the random variable specified by the algorithm \mathcal{A} and the function $v(x)$. The task is to obtain the expectation value

$$\mathbb{E}[v(\mathcal{A})] := \sum_{x=0}^{2^n-1} |a_x|^2 v(x). \quad (16)$$

In addition, assume we can implement a rotation onto an ancilla qubit,

$$\mathcal{R}|x\rangle|0\rangle = |x\rangle(\sqrt{1-v(x)}|0\rangle + \sqrt{v(x)}|1\rangle). \quad (17)$$

These elements are now combined into a simple quantum algorithm to obtain the expectation value. First apply the algorithm \mathcal{A} :

$$\mathcal{A}|0^n\rangle = \sum_{x=0}^{2^n-1} a_x |x\rangle, \quad (18)$$

where $|0^n\rangle$ denotes the n qubit register with all qubits in the state $|0\rangle$. Then perform the rotation of an ancilla via \mathcal{R}

$$\begin{aligned} & \sum_{x=0}^{2^n-1} a_x |x\rangle|0\rangle \\ & \rightarrow \sum_{x=0}^{2^n-1} a_x |x\rangle(\sqrt{1-v(x)}|0\rangle + \sqrt{v(x)}|1\rangle) =: |\chi\rangle. \end{aligned} \quad (19)$$

Combining the two operations defines a unitary \mathcal{F} and the resulting state $|\chi\rangle$

$$\mathcal{F}|0^{n+1}\rangle := \mathcal{R}(\mathcal{A} \otimes \mathcal{I}_2)|0^{n+1}\rangle = |\chi\rangle. \quad (20)$$

Here, \mathcal{I}_d is the d -dimensional identity operator. Measuring the ancilla in the state $|1\rangle$ obtains as the success probability the expectation value

$$\mu := \langle \chi | (\mathcal{I}_{2^n} \otimes |1\rangle\langle 1|) | \chi \rangle = \sum_{x=0}^{2^n-1} |a_x|^2 v(x) \equiv \mathbb{E}[v(\mathcal{A})]. \quad (21)$$

This success probability can be obtained by repeating the procedure t times and collecting the clicks for the $|1\rangle$ state as a fraction of the total measurements. The variance is $\epsilon^2 = \frac{\mu(1-\mu)}{t}$ from the Bernoulli distribution, i.e. the standard deviation is $\epsilon = \sqrt{\frac{\mu(1-\mu)}{t}}$. Hence, the experiment has to be repeated

$$t = \mathcal{O}\left(\frac{\mu(1-\mu)}{\epsilon^2}\right) \quad (22)$$

times for a given accuracy ϵ . This quadratic dependency in ϵ is analogous to the classical Monte Carlo dependency Eq. (15). Obtaining a quadratic speedup for the number of repetitions is the core task of amplitude estimation.

The main tool to obtain a quantum speedup is to connect the desired expectation value to an eigenfrequency of an oscillating quantum system and then use another quantum degree of freedom (such as another register of qubits) as a probe to extract the eigenfrequency. Note that we can slightly redefine the quantity being measured. Define the unitary

$$\mathcal{V} := \mathcal{I}_{2^{n+1}} - 2\mathcal{I}_{2^n} \otimes |1\rangle\langle 1|, \quad (23)$$

for which $\mathcal{V} = \mathcal{V}^\dagger$ and $\mathcal{V}^2 = \mathcal{I}_{2^{n+1}}$. A measurement of \mathcal{V} on $|\chi\rangle$ obtains $\langle \chi | \mathcal{V} | \chi \rangle = 1 - 2\mu$. From this measurement we can extract the desired expectation value.

Any quantum state in the $(n+1)$ -qubit Hilbert space can be expressed as a linear combination of $|\chi\rangle$ and a specific orthogonal complement $|\chi^\perp\rangle$. Thus, we can express $\mathcal{V}|\chi\rangle = \cos(\theta/2)|\chi\rangle + e^{i\phi}\sin(\theta/2)|\chi^\perp\rangle$, with the

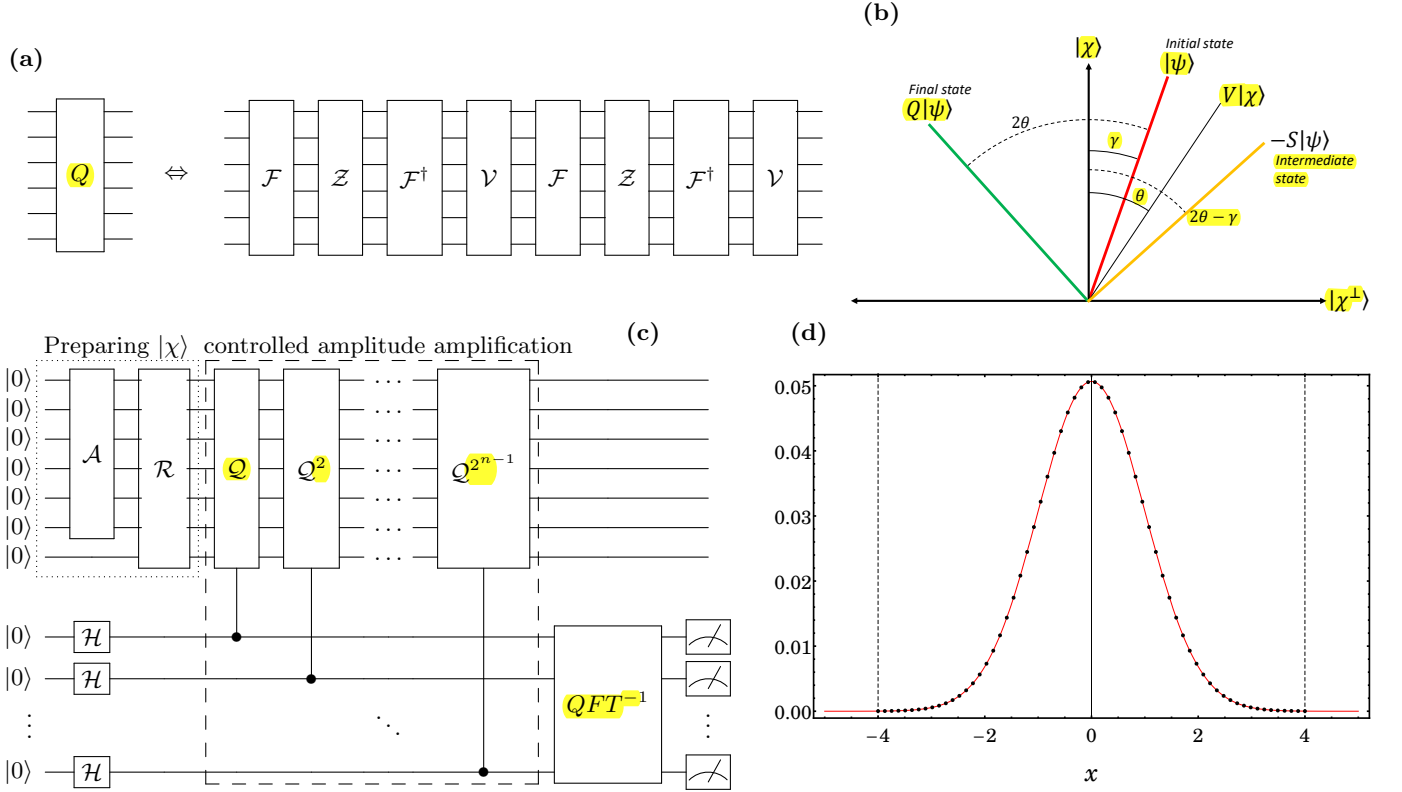


FIG. 2. Using amplitude estimation for the quantum Monte Carlo pricing of financial derivatives. (a) The $n+1$ qubit phase estimation unitary is written in terms of $\mathcal{F} := \mathcal{R}(\mathcal{A} \otimes \mathcal{I}_2)$, and the simple rotation unitaries $\mathcal{Z} := \mathcal{I}_{2^{n+1}} - 2|0^{n+1}\rangle\langle 0^{n+1}|$ and $\mathcal{V} := \mathcal{I}_{2^{n+1}} - 2\mathcal{I}_{2^n} \otimes |1\rangle\langle 1|$. (b) A visualization of the action of $\mathcal{Q} := \mathcal{U}\mathcal{S}$, with $\mathcal{S} = \mathcal{V}\mathcal{U}\mathcal{V}$ and $\mathcal{U} = \mathcal{F}\mathcal{Z}\mathcal{F}^\dagger$, on an arbitrary state $|\psi\rangle$ (red) in the span of $|\chi\rangle$ and $\mathcal{V}|\chi\rangle$. First, the action of $-\mathcal{S}$ on $|\psi\rangle$ is to reflect along $\mathcal{V}|\chi\rangle$, resulting in the intermediate $-\mathcal{S}|\psi\rangle$ (amber). Then, $-\mathcal{U}$ acts on $-\mathcal{S}|\psi\rangle$ by reflecting along $|\chi\rangle$. The resultant state $\mathcal{Q}|\psi\rangle$ (green) has been rotated anticlockwise by an angle 2θ in the hyperplane of $|\chi\rangle$ and $|\chi^\perp\rangle$. (c) The phase estimation circuit. Here, \mathcal{A} encodes the randomness by preparing a superposition in $|x\rangle$, while \mathcal{R} encodes the random variable into the $|1\rangle$ state of an ancilla qubit according to Eq. (17). The output after both steps is the multiqubit state $|\chi\rangle$. Amplitude estimation then proceeds by invoking phase estimation to encode the rotation angle θ in a register of quantum bits that are measured to obtain the estimate $\hat{\theta}$. (d) For pricing a European call option, the superposition prepared by \mathcal{A} (or equivalently \mathcal{G} in Eq. (35)) is a discretization of the normal distribution in x with a fixed cutoff (e.g. $c = 4$), approximating the Brownian motion of the underlying asset. In this case, \mathcal{R} encodes the call option payoff in Eq. (5).

angles ϕ and θ . Note that our expectation value can be retrieved via

$$1 - 2\mu = \cos(\theta/2). \quad (24)$$

The task becomes to measure θ . We now define a transformation \mathcal{Q} that encodes θ in its eigenvalues. First, define the unitary reflection

$$\mathcal{U} := \mathcal{I}_{2^{n+1}} - 2|\chi\rangle\langle\chi|, \quad (25)$$

which acts as $\mathcal{U}|\chi\rangle = -|\chi\rangle$ and $\mathcal{U}|\chi^\perp\rangle = |\chi^\perp\rangle$ for any orthogonal state. Note that $-\mathcal{U}$ reflects across $|\chi\rangle$ and leaves $|\chi\rangle$ itself unchanged. This unitary can be implemented as $\mathcal{U} = \mathcal{F}\mathcal{Z}\mathcal{F}^\dagger$, where \mathcal{F}^\dagger is the inverse of \mathcal{F} and $\mathcal{Z} := \mathcal{I}_{2^{n+1}} - 2|0^{n+1}\rangle\langle 0^{n+1}|$ is the reflection of the computational zero state. Similarly, define the unitary

$$\mathcal{S} := \mathcal{I}_{2^{n+1}} - 2\mathcal{V}|\chi\rangle\langle\chi|\mathcal{V} \equiv \mathcal{V}\mathcal{U}\mathcal{V}. \quad (26)$$

Note that $-\mathcal{S}$ reflects across $\mathcal{V}|\chi\rangle$ and leaves $\mathcal{V}|\chi\rangle$ itself unchanged. The transformation

$$\mathcal{Q} := \mathcal{U}\mathcal{S} = \mathcal{U}\mathcal{V}\mathcal{U}\mathcal{V} \quad (27)$$

performs a rotation by an angle 2θ in the two-dimensional Hilbert space spanned by $|\chi\rangle$ and $\mathcal{V}|\chi\rangle$. Figure 2 (a) shows the breakdown of \mathcal{Q} into its constituent unitaries and Fig. 2 (b) illustrates how \mathcal{Q} imprints a phase of 2θ via the reflections just discussed. The eigenvalues of \mathcal{Q} are $e^{\pm i\theta}$ with corresponding eigenstates $|\psi_\pm\rangle$ [15]. The task is to resolve these eigenvalues via phase estimation, as shown in Fig. 2 (c).

For phase estimation of θ [32], we require the conditional application of the operation \mathcal{Q} . Concretely, we require

$$\mathcal{Q}^c : |j\rangle|\psi\rangle \rightarrow |j\rangle\mathcal{Q}^j|\psi\rangle, \quad (28)$$

for an arbitrary n qubit state $|\psi\rangle$. Phase estimation then proceeds in the following way, see Fig. 2 (c). Take a copy of $|\chi\rangle$ by applying \mathcal{F} to a register of qubits in $|0^{n+1}\rangle$. Then prepare an additional m -qubit register in the uniform superposition via the Hadamard operation \mathcal{H}

$$\mathcal{H}^{\otimes m}|0^m\rangle|\chi\rangle = \frac{1}{\sqrt{2^m}} \sum_{j=0}^{2^m-1} |j\rangle|\chi\rangle. \quad (29)$$

Then perform the controlled operation \mathcal{Q}^c to obtain

$$\frac{1}{\sqrt{2^m}} \sum_{j=0}^{2^m-1} |j\rangle \mathcal{Q}^j |\chi\rangle. \quad (30)$$

One can show that $|\chi\rangle = \frac{1}{\sqrt{2}}(|\psi_+\rangle + |\psi_-\rangle)$ is the expansion of $|\chi\rangle$ into the two eigenvectors of \mathcal{Q} corresponding to the eigenvalues $e^{\pm i\theta}$ [15]. An inverse quantum Fourier transformation applied to Eq. (30) prepares the state

$$\sum_{x=0}^{2^m-1} \alpha_+(x)|x\rangle|\psi_+\rangle + \alpha_-(x)|x\rangle|\psi_-\rangle. \quad (31)$$

The $|\alpha_{\pm}(x)|^2$ are peaked where $x/2^m = \pm\hat{\theta}$ is an m -bit approximation to $\pm\theta$. Hence, measurement of the $|x\rangle$ register will retrieve the approximations $\pm\hat{\theta}$. The detailed steps are shown in Appendix F.

These results can be formalized with the following theorems.

Theorem 1 (Amplitude estimation [10]). *There is a quantum algorithm called amplitude estimation which takes as input: one copy of a quantum state $|\chi\rangle$, a unitary transformation $\mathcal{U} = \mathcal{I} - 2|\chi\rangle\langle\chi|$, a unitary transformation $\mathcal{V} = \mathcal{I} - 2P$ for some projector P , and an integer t . The algorithm outputs \hat{a} , an estimate of $a = \langle\chi|P|\chi\rangle$, such that*

$$|\hat{a} - a| \leq 2\pi \frac{\sqrt{a(1-a)}}{t} + \frac{\pi^2}{t^2}$$

with probability at least $8/\pi^2$, using \mathcal{U} and \mathcal{V} t times each.

This theorem can be used to estimate expectation values. Given the cosine relationship in Eq. (24), it is natural to start with $[0, 1]$ bounded expectation values.

Theorem 2 (Mean estimation for $[0, 1]$ bounded functions [22]). *Let there be given a quantum circuit \mathcal{A} on n qubits. Let $v(\mathcal{A})$ be the random variable that maps to $v(x) \in [0, 1]$ when the bit string x is measured as the output of \mathcal{A} . Let \mathcal{R} be defined as*

$$\mathcal{R}|x\rangle|0\rangle = |x\rangle(\sqrt{1-v(x)}|0\rangle - \sqrt{v(x)}|1\rangle).$$

Let $|\chi\rangle$ be defined as $|\chi\rangle = \mathcal{R}(\mathcal{A} \otimes \mathcal{I}_2)|0^{n+1}\rangle$. Set $\mathcal{U} = \mathcal{I}_{2^{n+1}} - 2|\chi\rangle\langle\chi|$. There exists a quantum algorithm

that uses $\mathcal{O}(\log 1/\delta)$ copies of the state $|\chi\rangle$, uses \mathcal{U} for a number of times proportional to $\mathcal{O}(t \log 1/\delta)$ and outputs an estimate $\hat{\mu}$ such that

$$|\hat{\mu} - E[v(\mathcal{A})]| \leq C \left(\frac{\sqrt{\mathbb{E}[v(\mathcal{A})]}}{t} + \frac{1}{t^2} \right)$$

with probability at least $1 - \delta$, where C is a universal constant. In particular, for any fixed $\delta > 0$ and any ϵ such that $0 < \epsilon \leq 1$, to produce an estimate $\hat{\mu}$ such that with probability at least $1 - \delta$, $|\hat{\mu} - E[v(\mathcal{A})]| \leq \epsilon E[v(\mathcal{A})]$, it suffices to take $t = \mathcal{O}\left((1/(\epsilon\sqrt{E[v(\mathcal{A})])}\right)$. To achieve $|\hat{\mu} - E[v(\mathcal{A})]| \leq \epsilon$ with probability at least $1 - \delta$, it suffices to take $t = \mathcal{O}(1/\epsilon)$.

This theorem is a direct application of amplitude estimation via Theorem 1. The success probability of $8/\pi^2$ of amplitude estimation can be improved to $1 - \delta$ by taking the median of multiple runs of Theorem 1, see Appendix F, Lemma 6. Theorem 2 can be generalized to random variables with bounded variance as follows.

Theorem 3 (Mean estimation with bounded variance [22]). *Let there be given a quantum circuit \mathcal{A} . Let $v(\mathcal{A})$ be the random variable corresponding to $v(x)$ when the outcome x of \mathcal{A} is measured, such that $\mathbb{V}[v(\mathcal{A})] \leq \lambda^2$. Let the accuracy be $\epsilon < 4\lambda$. Take \mathcal{U} and $|\chi\rangle$ as in Theorem 2. There exists a quantum algorithm that uses $\mathcal{O}(\log(\lambda/\epsilon) \log \log(\lambda/\epsilon))$ copies of $|\chi\rangle$ and uses \mathcal{U} for a number of times $\mathcal{O}\left((\lambda/\epsilon) \log^{3/2}(\lambda/\epsilon) \log \log(\lambda/\epsilon)\right)$ and estimates $\mathbb{E}[v(\mathcal{A})]$ up to additive error ϵ with success probability at least $2/3$.*

Theorem 3 can be proved by employing Theorem 2. We proceed by sketching the proof, and refer the interested reader to the detailed treatment in [22]. To show Theorem 3, the bounded-variance random variable $v(\mathcal{A})$ is related to a set of random variables with outputs between $[0, 1]$ and then the estimates of the mean of each of these random variables are combined to give the final estimate. This can be done in three steps. First, the random variable $v(\mathcal{A})$ can be approximately standardized by subtracting an approximation to the mean and dividing by the known variance bound λ^2 , to obtain a random variable $v'(\mathcal{A})$. Second, $v'(\mathcal{A})$ can be split into positive and negative parts by using the functions $f_{\min}(x) = \min\{x, 0\}$ and $f_{\max}(x) = \max\{x, 0\}$. (Coincidentally, these function are similar to the call and put option payoff functions.) This defines new random variables $B_{<} = -f_{\min}(v'(\mathcal{A}))$ and $B_{>} = f_{\max}(v'(\mathcal{A}))$, both taking on only values ≥ 0 . These random variables can be rescaled and combined to give the desired random variable.

As the third step, both positive random variables $B_{<}, B_{>}$ can be split into multiple auxiliary random variables with outputs between $[0, 1]$ and each of these random variables is estimated by Theorem 2. This is

done by defining the functions for $0 \leq a < b$

$$f_{a,b}(x) = \frac{1}{b} \begin{cases} x & \text{if } a \leq x < b, \\ 0 & \text{otherwise.} \end{cases} \quad (32)$$

which take on values in $[0, 1]$. Now the auxiliary random variables $f_{0,1}(B)$, $f_{1,2}(B)$, $f_{2,4}(B)$, $f_{4,8}(B)$ and so forth can be defined which are all taking values in $[0, 1]$. Theorem 2 can be used to estimate the mean of each of these random variables. It can be shown that only a small number $\lceil \log(\lambda/\epsilon) \rceil$ of these auxiliary random variables are needed to estimate the mean of random variable B with final error ϵ . This estimation makes use of the bounded-variance property which leads to the distribution tails contributing only a small error.

The resource count of Theorem 3 is justified as follows. For an estimation with accuracy ϵ it can be shown that the number of random variables $f_{a,b}(B)$ and therefore the number of applications of Theorem 2 needed is $\lceil \log(\lambda/\epsilon) \rceil$. In addition, the number of steps t in Theorem 2 can be taken to be $t = \mathcal{O}\left((\lambda/\epsilon)\sqrt{\log \lambda/\epsilon}\right)$ and also $\delta = \mathcal{O}(1/\log(\lambda/\epsilon))$ to achieve the final accuracy. Thus, from each application of the Theorem 2, we need $\mathcal{O}(\log \log(\lambda/\epsilon))$ copies of $|\chi\rangle$ and $\mathcal{O}\left(\frac{\lambda}{\epsilon}\sqrt{\log(\lambda/\epsilon)}\log \log(\lambda/\epsilon)\right)$ applications of \mathcal{U} . As we apply Theorem 2 for $\lceil \log(\lambda/\epsilon) \rceil$ times we require $\mathcal{O}(\log(\lambda/\epsilon)\log \log(\lambda/\epsilon))$ copies of $|\chi\rangle$ and $\mathcal{O}\left(\frac{\lambda}{\epsilon}(\log(\lambda/\epsilon))^{3/2}\log \log(\lambda/\epsilon)\right)$ applications of \mathcal{U} . This is an almost quadratic speedup in the number of applications of \mathcal{U} when compared to the direct approach outlined in Eq. (22).

V. QUANTUM ALGORITHM FOR EUROPEAN OPTION PRICING

We specialize the above discussion to the Monte Carlo pricing of a European call option. We show how to prepare the Brownian motion distribution and the quantum circuit for the option payoff. For any European option, i.e. an option that depends on the asset prices only at a single maturity date T , we can write the price as an expectation value of a function of the underlying stochastic processes evaluated at the maturity date. For the BSM model with a single Brownian motion we have

$$\Pi = e^{-rT} \mathbb{E}_{\mathbb{Q}}[v(W_T)], \quad (33)$$

where W_T is the Brownian motion at time T and $v(x)$ is, for example, defined from the function $f(x)$ in Eq. (5). From Definition 1, W_T is a Gaussian random variable $\sim \mathcal{N}(0, T)$. The probability density for this random variable is given by

$$p_T(x) = \frac{1}{\sqrt{2\pi T}} e^{-\frac{x^2}{2T}}. \quad (34)$$

To prepare an approximate superposition of these probabilities, we take the support of this density from

$[-\infty, \infty] \rightarrow [-x_{\max}, x_{\max}]$ and discretize this interval with 2^n points, where n is an integer. Here, $x_{\max} = \mathcal{O}(\sqrt{T})$, as a few standard deviations are usually enough to capture the normal distribution and reliably estimate the options price. The discretization points may be defined as $x_j := -x_{\max} + j\Delta x$, with $\Delta x = 2x_{\max}/(2^n - 1)$ and $j = 0, \dots, 2^n - 1$. Define the probabilities $p_j = p_T(x_j)/C$, with the normalization $C = \sum_{j=0}^{2^n-1} p_T(x_j)$. This process is illustrated in Fig. 2 (d). According to Ref. [33], there exists a quantum algorithm \mathcal{G} (which takes on the role of \mathcal{A} in the previous section) such that we can prepare

$$\mathcal{G}|0^n\rangle = \sum_{j=0}^{2^n-1} \sqrt{p_j}|j\rangle. \quad (35)$$

This algorithm runs in $\mathcal{O}(n)$ steps, provided that there is a method to efficiently sample the integrals $\int_a^b p_T(x)dx$ for any a, b . These integrals can be efficiently sampled for any log-concave distribution, such as in the present case of the Gaussian distribution associated with W_T . We show the steps in the Appendix E.

Now consider the function $v(x) : \mathbb{R} \rightarrow \mathbb{R}$ which relates the Brownian motion to the option payoff. For the example of the European call option, the function is

$$v_{\text{euro}}(x) = \max\{0, S_0 e^{\sigma x + (r - \frac{1}{2}\sigma^2)T} - K\}. \quad (36)$$

At the discretization points of the Brownian motion we define

$$v(j) := v(x_j). \quad (37)$$

One can find a binary approximation to this function over n bits, i.e. $\tilde{v}(j) : \{0, 1\}^n \rightarrow \{0, 1\}^n$, where we take the number of input bits to be the same as the number of output bits. The n bits allow one to represent 2^n different floating points numbers, and with $n = n_1 + n_2$ one can trade off the largest represented number 2^{n_1} with the accuracy of the representation, 2^{-n_2} [34]. We can take $n = n_2$, by keeping track of the exponent of the floating point numbers offline. The accuracy for the function approximation is given by $|v(j) - \tilde{v}(j)| = \mathcal{O}(1/2^n)$, if $v(j)$ is sufficiently well behaved (e.g. Lipschitz continuous.) The classical circuit depth of most such functions $\tilde{v}(j)$ discretizing real-world options payoffs is $\mathcal{O}(n)$. Via the reversible computing paradigm, this classical circuit can be turned into a reversible classical circuit using $\mathcal{O}(n)$ operations [32]. Given the reversible classical circuit, the quantum circuit can be determined involving $\mathcal{O}(n)$ quantum gates. In other words we can implement the operation

$$|j\rangle|0^n\rangle \rightarrow |j\rangle|\tilde{v}(j)\rangle. \quad (38)$$

See Appendix C for more details on basic arithmetic operations and quantum circuits for options payoffs.

We can now go through the steps of the quantum Monte Carlo algorithm. Sec. IV assumes availability of \mathcal{R} with a real-valued function $v(x)$. Such a rotation can be directly implemented in some cases [35]. Here, we invoke the controlled rotation $\mathcal{R}|j\rangle|0\rangle = |j\rangle\left(\sqrt{1-\tilde{v}(x_j)}|0\rangle + \sqrt{\tilde{v}(x_j)}|1\rangle\right)$, with the discretized options payoff function $\tilde{v}(x)$. See Appendix D for an implementation of this rotation by using an auxiliary register of qubits and the circuit for the options payoff. The steps are similar to before,

$$\mathcal{G}|0^n\rangle = \sum_{j=0}^{2^n-1} \sqrt{p(x_j)}|j\rangle \quad (39)$$

$$\rightarrow \sum_{j=0}^{2^n-1} \sqrt{p(x_j)}|j\rangle \quad (40)$$

$$\left(\sqrt{1-\tilde{v}(x_j)}|0\rangle + \sqrt{\tilde{v}(x_j)}|1\rangle\right) =: |\chi\rangle.$$

Measuring the ancilla in the state $|1\rangle$, we obtain the expectation value

$$\mu = \langle\chi|(\mathcal{I}_{2^n} \otimes |1\rangle\langle 1|)|\chi\rangle = \sum_{j=0}^{2^n-1} p_T(x_j)\tilde{v}(x_j). \quad (41)$$

This expectation value μ , assuming it can be measured exactly, determines the option price $\mathbb{E}_{\mathbb{Q}}[v(W_T)]$ to accuracy

$$|\mu - \mathbb{E}_{\mathbb{Q}}[v(W_T)]| =: \nu. \quad (42)$$

The error arises from the discretization of the probability density and the accuracy of the function approximation \tilde{v} . Using n qubits the accuracy is given by $\nu = \mathcal{O}(2^{-n})$.

We can employ phase estimation and Theorems 2 and 3 to evaluate μ to a given accuracy and get a bound on the number of computational steps needed. For the European call option, we can show that the variance is bounded by $\mathbb{V}_{\mathbb{Q}}[f(S_T)] \leq \lambda^2$ where $\lambda^2 := \mathcal{O}(\text{poly}(S_0, e^{rT}, e^{\sigma^2 T}, K))$, see Appendix A. Thus from Theorem 3 we know that we can use $\mathcal{O}(\log(\lambda/\epsilon) \log \log(\lambda/\epsilon))$ copies of $|\chi\rangle$ and $\mathcal{O}((\lambda/\epsilon) \log^{3/2}(\lambda/\epsilon) \log \log(\lambda/\epsilon))$ applications of \mathcal{U} to provide an estimate $\hat{\mu}$ for μ up to additive error ϵ with success probability at least $2/3$. The accuracy is $\epsilon < 4\lambda$. The total error is

$$|\hat{\mu} - \mathbb{E}_{\mathbb{Q}}[f(S_T)]| \leq \epsilon + \nu, \quad (43)$$

compounding the two sources from amplitude estimation and the discretization error. Discounting $\hat{\mu}$, see Eq. (33), then retrieves an estimation of the option price $\hat{\Pi}$. The total number of applications of \mathcal{U} is

$$\tilde{\mathcal{O}}\left(\frac{\lambda}{\epsilon}\right), \quad (44)$$

where $\tilde{\mathcal{O}}(\cdot)$ suppresses polylogarithmic factors. This quantity can be considered the analogue of the classical number of Monte Carlo runs [15, 22]. The required number of quantum steps is quadratically better than the classical number of steps in Eq. (15), or the naive quantum case in Eq. (22).

VI. ASIAN OPTION PRICING

Up to this point, we have discussed the quantum Monte Carlo pricing of derivatives via the illustrative example of the European call option. This call option can be priced analytically in the Black-Scholes-Merton framework, thus MC methods are in principle not required. Another family of options is the so-called Asian options, which depend on the average asset price before the maturity date [30].

Definition 4 (Asian options). *The Asian call option payoff is defined as*

$$f(A_T) = \max\{0, A_T - K\}, \quad (45)$$

where K is the strike price and T the maturity date. The arithmetic mean option value is defined via

$$A_T^{\text{arith}} = \frac{1}{L} \sum_{l=1}^L S_{t_l}, \quad (46)$$

and the geometric mean option is defined via

$$A_T^{\text{geo}} = \exp \frac{1}{L} \sum_{l=1}^L \log S_{t_l}, \quad (47)$$

for pre-defined time points $0 < t_1 < \dots < t_L \leq T$, with $L \geq 1$.

The following discussion assumes the BSM framework as before. In this framework, the geometric mean Asian option can be priced analytically, while such a solution is not known for the arithmetic Asian option. Assume for this discussion that all adjacent time points are separated by the interval Δt , i.e. $t_{l+1} - t_l = \Delta t = T/L$ for all $l = 1, \dots, L-1$. Analogously to before, we can efficiently prepare via the Grover-Rudolph algorithm [33] a state that corresponds to the Gaussian normal distribution with variance Δt

$$|p_{\Delta t}\rangle := \mathcal{G}|0^m\rangle = \sum_{j=0}^{2^m-1} \sqrt{p_{\Delta t}(x_j)}|j\rangle \quad (48)$$

This state uses m qubits and takes $\mathcal{O}(m)$ steps to prepare. Then we prepare the product state with L such states, i.e.,

$$|p\rangle := |p_{\Delta t}\rangle \dots |p_{\Delta t}\rangle. \quad (49)$$

This state uses Lm qubits and takes $\mathcal{O}(Lm)$ steps to prepare.

In addition, we require the operation

$$|j_1, \dots, j_L\rangle|0\rangle = |j_1, \dots, j_L\rangle|A(S_{t_1}(x_{j_1}), \dots, S_{t_L}(x_{j_L}))\rangle. \quad (50)$$

Here, $A(S_{t_1}(x_{j_1}), \dots, S_{t_L}(x_{j_L}))$ is the average stock price corresponding to the Brownian path x_{j_1}, \dots, x_{j_L} . This operation is easily computable. Each index j is mapped to its corresponding point x_j via $x_j = -x_{\max} + j\Delta x$ as before. Then start at the known S_0 and use

$$S_{t_{i+1}}(x) = S_{t_i}e^{\sigma x + (r - \sigma^2/2)\Delta t} \quad (51)$$

to obtain the stock price at the next time point, where x is a sample of the Brownian motion. This step can also be performed in the log domain [30]

$$\log S_{t_{i+1}}(x) = \log S_{t_i} + \sigma x + (r - \sigma^2/2)\Delta t. \quad (52)$$

In this way, one obtains a state where the label $|j_1, \dots, j_L\rangle$ associated with the corresponding stock price path,

$$|j_1, \dots, j_L\rangle|S_{t_1}(x_{j_1})\rangle \dots |S_{t_L}(x_{j_L})\rangle. \quad (53)$$

Moreover, the average (both arithmetic and geometric) can be computed in a sequential manner, since we can implement the step

$$|j_1, \dots, j_L\rangle|S_{t_i}(x_{j_i})\rangle|A(S_{t_1}(x_{j_1}), \dots, S_{t_i}(x_{j_i}))\rangle \rightarrow (54) \\ |j_1, \dots, j_L\rangle|S_{t_{i+1}}(x_{j_{i+1}})\rangle|A(S_{t_1}(x_{j_1}), \dots, S_{t_{i+1}}(x_{j_{i+1}}))\rangle.$$

The steps are performed until the final time t_L is reached and $A(S_{t_1}(x_{j_1}), \dots, S_{t_L}(x_{j_L}))$ is stored in a register of qubits. Reversibility of the quantum arithmetic operations guarantees that registers storing the intermediate steps can be uncomputed. Applying operation Eq. (50) to the product state Eq. (49) obtains

$$\sum_{j_1 \dots j_L=0}^{2^m-1} \sqrt{p_{j_1, \dots, j_L}} |j_1, \dots, j_L\rangle |A(S_{t_1}(x_{j_1}), \dots, S_{t_L}(x_{j_L}))\rangle. \quad (55)$$

with $\sqrt{p_{j_1, \dots, j_L}} := \sqrt{p_{\Delta t}(x_{j_1})} \dots \sqrt{p_{\Delta t}(x_{j_L})}$. Analogously to before, a conditional rotation of an ancilla qubit can be performed such that measuring the ancilla in the $|1\rangle$ state obtains

$$\sum_{j_1 \dots j_L=0}^{2^m-1} p_{j_1, \dots, j_L} f(A(S_{t_1}(x_{j_1}), \dots, S_{t_L}(x_{j_L}))) \approx \mathbb{E}_{\mathcal{Q}}[f(A)]. \quad (56)$$

The result is an approximation to the Black-Scholes price of the Asian option. The variance of the option can be bounded from the fact that the arithmetic mean upper bounds the geometric mean and the arithmetic mean itself is upper bounded by the expected maximum of the stock price $\max\{S_{t_1}, \dots, S_{t_L}\}$. The variance of options such as calls or puts on the maximum of a stock in a time period can be bounded [2] similar to the variance bound of the European call option, which is presented in Appendix A. We thus obtain a similar speedup for the Asian options via Theorem 3.

VII. NUMERICAL SIMULATIONS

While a practical quantum computer has yet to become a reality, we can exhibit the speedup of our quantum algorithm for options pricing numerically, and compare its performance with the classical Monte Carlo method. Note that our quantum algorithm consists of two main parts, see also Fig. 2 (c). First, prepare the Brownian motion superposition (through \mathcal{A}) and encode the option payoff onto an ancilla qubit (through \mathcal{R}); and second, use amplitude amplification and phase estimation with repeated applications of \mathcal{Q} to estimate the expectation value encoded in the ancilla qubit. The phase estimation subroutine can in principle be simulated using publicly available quantum software packages such as Strawberry Fields [36] and ProjectQ [37]. However, to showcase the quadratic speed up, we here perform phase estimation by using a single qubit rotated according to $e^{i\theta\sigma_z/2}$, where θ is the predetermined phase.

We perform numerics for a phase θ given by the European call option, see Sec. II, which can be priced analytically. The analytical price Π is computed directly from the Black-Scholes-Merton formula, see Result 1. We provide an estimate $\hat{\Pi}$ using both quantum phase estimation and the standard classical Monte Carlo method. Here, the analytical price Π is used both as an input to the single-qubit phase estimation via θ from Eq. (24), as well as a benchmark for the resultant simulations. The single-qubit phase estimation is described in Appendix F. We define the corresponding estimation error as the difference between the estimated price $\hat{\Pi}$ and analytical price Π , i.e.

$$\text{Error} := |\hat{\Pi} - \Pi|. \quad (57)$$

In the figures and following discussion, we will use subscripts Q and C to denote the quantum and classical estimations respectively. The estimation error follows a power-law behavior with the number of MC steps k undertaken

$$\text{Error} = a k^\zeta, \quad (58)$$

where ζ is the scaling exponent and a is a constant. As discussed in Sec. III, as well as obtained in our simulations, the scaling exponent for classical MC estimation is $\zeta_C = -1/2$. For the quantum case, the k_Q is the total number of applications of the single-qubit unitary. The simulations are performed such that quantum and classical estimates have a similar confidence ($> 99.5\%$), which implies a number of independent single-qubit phase estimation runs of $D \approx 24$ [15], see also Appendix F.

Fig. 3 shows a comparison between the error scalings for our quantum algorithm (blue solid curve with markers) and classical MC (orange dashed curve). The parameters for this figure are: $S_0 = \$100$, $K = \$50$, $r = 0.05$, $\sigma = 0.2$, $T = 1$, and $D = 24$. The analytical price was determined to be $\Pi = \$10.5$ and rescaled to

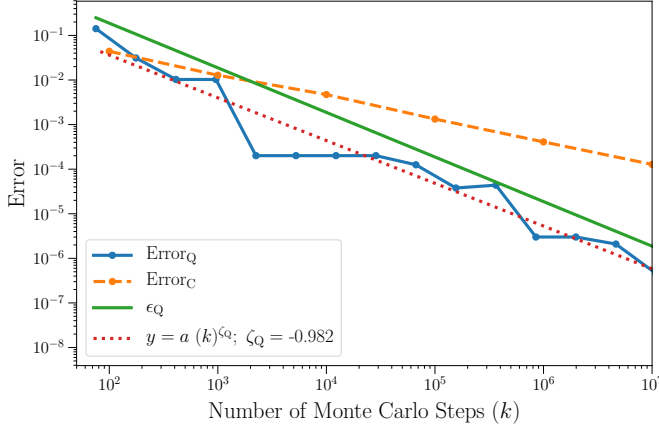


FIG. 3. Scaling of the error in classical and quantum MC methods (defined in Eq. (57)) plotted against number of MC steps for a European call option in log-log scale with $S_0 = \$100$, $K = \$50$, $r = 0.05$, $\sigma = 0.2$, $T = 1$, and $D = 24$. Subscripts C and Q denote the errors from classical MC and quantum phase estimation, respectively. Evidently, the error for the quantum algorithm (with a fitted slope of $\zeta_Q = -0.982$) scales almost quadratically faster than the classical MC method (which has $\zeta_C = -0.5$). The theoretical upper bound on the error in quantum algorithm is shown by the solid green curve, which corresponds to $\zeta_Q = -1$.

the corresponding $\theta = 2 \arccos(1 - 2\Pi/S_0)$ via Eq. (24) for use in the phase estimation. In addition, we have also plotted a fit of the quantum error to the power law given in Eq. (58) resulting in the scaling exponent to be $\zeta_Q = -0.982$. It is evident that the scaling exponent of quantum phase estimation is almost twice the classical one. Indeed, the green solid curve shows the upper bound on the errors in quantum estimation defined as [15], see also Appendix F:

$$\epsilon_Q := \left| \cos\left(\frac{\hat{\theta}}{2} + \frac{\pi}{k_Q}\right) - \cos\left(\frac{\hat{\theta}}{2}\right) \right|, \quad (59)$$

where $\hat{\theta}$ is the phase estimated via the quantum algorithm. This upper bound has a scaling exponent of -1 and hence straightforwardly demonstrates the quadratic speedup in the number of steps for phase estimation to a given error.

To test the robustness of the quadratic speedup in scaling, we vary the strike price and plot the ratio of the quantum to the classical scaling exponents ζ_Q/ζ_C in Fig. 4. We obtain an almost quadratic advantage in estimation overhead for all strike prices. Similar tests by varying other parameters also show a robust quadratic quantum speedup of the Monte Carlo estimation.

VIII. DISCUSSION AND CONCLUSION

In this work, we have presented a quantum algorithm for the pricing of financial derivatives. We have

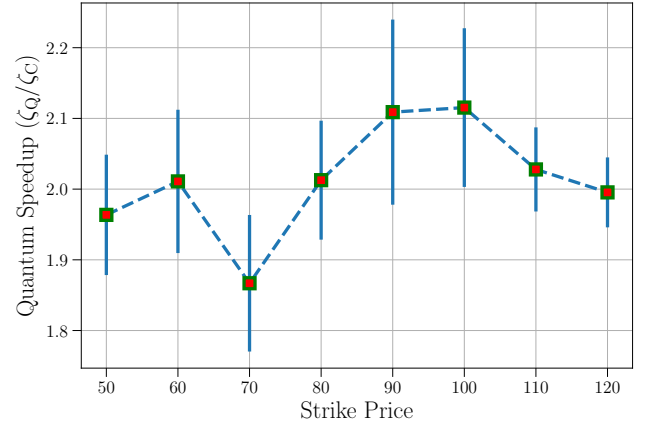


FIG. 4. Ratio of the quantum to classical scaling exponents. The quantum scaling is obtained by fitting the simulations results to the power law in Eq. (58), while the classical scaling exponent is taken to be -0.5 . Results are plotted with varying strike price K (in dollars) and fixing other parameters to be the same as in Fig. 3. An almost quadratic speed up is obtained for all chosen values of K .

assumed that the distribution of the underlying random variables, i.e. the martingale measure, is known and the corresponding quantum states can be prepared efficiently. In addition, we assume efficient computability of the derivative payoff function. Under these assumptions, we exhibit a quadratic speedup in the number of samples required to estimate the price of the derivative up to a given error: if the desired accuracy is ϵ , then classical methods show a $1/\epsilon^2$ dependency in the number of samples, while the quantum algorithm shows a $1/\epsilon$ dependency.

As an exemplary case, we have discussed European call options for which analytical solutions are known. In addition, we have discussed Asian options, which in the arithmetic averaging case require Monte Carlo methods to be priced. Our approach can in principle be applied to any derivative which has a payoff that is a function that can be efficiently broken down into elementary arithmetic operations within a quantum circuit, and which can also depend on an average over multiple time windows. Future work will extend these discussions to the complex payoff functions often used at leading financial institutions, as well as addressing more complicated stochastic models.

Monte Carlo simulations play a major role in managing the risk that a financial institution is exposed to [3]. Especially after the financial crisis of 2008-9, sophisticated risk management is increasingly important to banks internally and also required by government regulators [38, 39]. Such risk analysis falls under the umbrella of so-called valuation adjustments (VA), or XVA [38, 39], where X stands for the type of risk under consideration. An example is CVA, where the counterparty credit risk is modeled. Such a valuation adjusts the price of the derivative based on the risk that

the counterparty in that financial contract runs out of money.

XVA calculations are a major computational effort for groups ('desks') at financial institutions that handle complex derivatives such as those based on interest rates. For complex financial derivatives, such risk management involves a large amount of Monte Carlo simulations. Different Monte Carlo runs assess the price of a derivative under various scenarios. Determining the risk of the complete portfolio of a desk often requires overnight calculations of the prices according to various risk scenarios. Quantum computers promise a significant speedup for such computations. In principle, overnight calculations could be reduced to much shorter time scales (such as minutes), which would allow a more real time analysis of risk. Such close-to real time analysis would allow the institution to react faster to changing market conditions and to profit from trading opportunities.

The qubit model for universal quantum computing was employed here to provide a succinct discussion.

However, one is not restricted to such a setting but can use other universal quantum computational models such as adiabatic quantum computation or continuous variable (CV) quantum computation. In the continuous variable setting, instead of qubits one has oscillators which in principle have infinite dimensional Hilbert spaces. In this setting, preparing Gaussian states for the probability distributions can be done in a straightforward manner, instead of employing the relatively complicated preparation routine according to Grover-Rudolph. Investigating the promising advantages of the CV setting in a financial context in more detail will be left for future work.

ACKNOWLEDGMENTS

We acknowledge Juan Miguel Arrazola, John C. Hull, Ali Najmaie and Vikash Ranjan for insightful discussions.

-
- [1] J. C. Hull, *Options, futures, and other derivatives* (Prentice Hall, 2012).
 - [2] S. Shreve, *Stochastic Calculus for Finance II: Continuous-Time Models* (Springer-Verlag, 2004).
 - [3] H. Föllmer and A. Schied, *Stochastic Finance: An Introduction in Discrete Time* (Walter de Gruyter, 2004).
 - [4] F. Black and M. Scholes, *Journal of Political Economy* **81**, 637 (1973).
 - [5] R. C. Merton, *The Bell Journal of Economics and Management Science* **4**, 141 (1973).
 - [6] R. Eckhardt, *Los Alamos Science* **15**, 131 (1987).
 - [7] P. Glasserman, *Monte Carlo Methods in Financial Engineering* (Springer-Verlag, 2003).
 - [8] L. K. Grover, in *STOC '96 Proceedings of the 28th Annual ACM Symposium on the Theory of Computing* (ACM, New York, 1996), pp. 212–219.
 - [9] C. Dürr and P. Hoyer, arXiv preprint quant-ph/9607014 (1996).
 - [10] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, *Contemporary Mathematics* **305**, 53 (2002).
 - [11] S. Heinrich, *Journal of Complexity* **18**, 1 (2002).
 - [12] A. Ambainis, *SIAM Journal on Computing* **37**, 210 (2007).
 - [13] M. Szegedy, in *FOCS '04 Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science* (IEEE Computer Soc., Washington, D.C., 2004), pp. 32–41.
 - [14] P. Wocjan, C.-F. Chiang, D. Nagaj, and A. Abeyesinghe, *Phys. Rev. A* **80**, 022340 (2009).
 - [15] G. Xu, A. J. Daley, P. Givi, and R. D. Somma, *AIAA Journal* **56**, 687 (2018).
 - [16] V. Giovannetti, S. Lloyd, and L. Maccone, *Physical Review Letters* **96**, 010401 (2006).
 - [17] F. Magniez, A. Nayak, J. Roland, and M. Santha, in *STOC '07 Proceedings of the 39th Symposium on the Theory of Computing* (ACM, New York, 2007), pp. 575–584.
 - [18] E. Knill, G. Ortiz, and R. Somma, *Physical Review A* **75**, 012328 (2007).
 - [19] R. D. Somma, S. Boixo, H. Barnum, and E. Knill, *Physical Review Letters* **101**, 130504 (2008).
 - [20] D. Poulin and P. Wocjan, *Physical Review Letters* **102**, 130503 (2009).
 - [21] A. Chowdhury and R. D. Somma, *Quantum Information and Computation* **17**, 0041 (2017).
 - [22] A. Montanaro, *Proc. R. Soc. A* **471**, 0301 (2015).
 - [23] B. E. Baaquie, *Quantum finance* (Cambridge University Press, 2004).
 - [24] M. L. de Prado, *Advances in financial machine learning* (John Wiley & Sons, 2018).
 - [25] I. Halperin, arXiv preprint arXiv:1712.04609 (2017).
 - [26] G. Rosenberg, P. Haghnegahdar, P. Goddard, P. Carr, K. Wu, and M. L. de Prado, *IEEE Journal of Selected Topics in Signal Processing* **10**, 1053 (2016).
 - [27] N. Elsokkary, F. S. Khan, D. La Torre, T. S. Humble, and J. Gottlieb, Tech. Rep., Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States) (2017).
 - [28] G. Rosenberg, 1QBit white paper: <https://1qbit.com/whitepaper/arbitrage/> (2016).
 - [29] F. Longstaff and E. Schwartz, *Review of Financial Studies* **14**, 113 (2001).
 - [30] A. Kemna and A. Vorst, *Journal of Banking and Finance* **14**, 113 (1990).
 - [31] W. Schoutens, *Lévy Processes in Finance: Pricing Financial Derivatives* (Wiley, 2003).
 - [32] M. A. Nielsen and I. Chuang, *Quantum computation and quantum information* (Cambridge University Press, Cambridge, 2002).
 - [33] L. Grover and T. Rudolph, arXiv preprint quant-ph/0208112 (2002).
 - [34] IEEE standard for Floating-Point Arithmetic, IEEE Std 754-2008 pp. 1–70 (2008).
 - [35] G. H. Low and I. L. Chuang, *Phys. Rev. Lett.* **118**, 010501 (2017).

- [36] N. Killoran, J. Izaac, N. Quesada, V. Bergholm, M. Amy, and C. Weedbrook, arXiv preprint arXiv:1804.03159 (2018).
- [37] D. S. Steiger, T. Häner, and M. Troyer, Quantum **2**, 49 (2018).
- [38] A. Green, *XVA: Credit, Funding and Capital Valuation Adjustments* (John Wiley & Sons, 2015).
- [39] P. J. Zeitsch, Journal of Mathematical Finance **7**, 239 (2017).
- [40] V. Vedral, A. Barenco, and A. Ekert, Phys. Rev. A **54**, 147 (1996).
- [41] D. Beckman, A. N. Chari, S. Devabhaktuni, and J. Preskill, Phys. Rev. A **54**, 1034 (1996).
- [42] R. V. Meter and K. M. Itoh, Phys. Rev. A **71**, 052320 (2005).
- [43] T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore, Quantum Information and Computation **6**, 351 (2006).
- [44] Y. Takahashi, S. Tani, and N. Kunihiro, Quantum Information and Computation **10**, 0872 (2010).
- [45] M. K. Bhaskar, S. Hadfield, A. Papageorgiou, and I. Petras, Quantum Information and Computation **16**, 0197 (2016).
- [46] N. Wiebe and M. Roetteler, Quantum Information and Computation **16**, 134 (2016).
- [47] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **454**, 339 (2018).
- [48] D. Nagaj, P. Wocjan, and Y. Zhang, Quantum Information and Computation **9**, 1053 (2009).

Appendix A: Black-Scholes calculations

Here, we show several calculations related to the Black-Scholes-Merton model. We show the solution to the stochastic differential equation, the martingale property of the stock price, the BSM price for the European call option, and an upper bound for the variance of the BSM price for the call option. First note the following result.

Result 2. (informal) *For the infinitesimal Brownian increment it holds that*

$$dW_t^2 = dt \quad (\text{A1})$$

This result can be reasoned by the variance of the Brownian increment being proportional to the time interval of the increment. Next, we show the solution to the following stochastic differential equation.

Lemma 1. *The stochastic differential equation*

$$dS_t = S_t \alpha dt + S_t \sigma dW_t, \quad (\text{A2})$$

is solved by

$$S_t = S_0 e^{\sigma W_t + (\alpha - \frac{\sigma^2}{2})t}. \quad (\text{A3})$$

Proof. We show this result by finding the differential dS_t given S_t . The quantity S_t can be seen as a function

$s(t, x)$. To obtain the differential, expand this function to second order in dx and use the solution S_t to obtain

$$\begin{aligned} ds(t, x) &= \frac{\partial s}{\partial t} dt + \frac{\partial s}{\partial x} dx + \frac{1}{2} \frac{\partial^2 s}{\partial x^2} dx^2 + \mathcal{O}(dt^2, dx^3) \quad (\text{A4}) \\ &= (\alpha - \frac{\sigma^2}{2}) s(t, x) dt + \sigma s(t, x) dx + \frac{\sigma^2}{2} s(t, x) dx^2. \end{aligned}$$

Using $dx = dW_t$ and $dx^2 = dW_t^2 = dt$ from Result 2 leads to the differential

$$dS_t = S_t \alpha dt + S_t \sigma dW_t. \quad (\text{A5})$$

□

Next, we show the martingale property of the discounted stock price.

Lemma 2. *Under the \mathbb{Q} measure, the stock price is a martingale, i.e.*

$$S_0 = e^{-rT} \mathbb{E}_{\mathbb{Q}}[S_T]. \quad (\text{A6})$$

Proof.

$$e^{-rT} \mathbb{E}_{\mathbb{Q}}[S_T] = S_0 \mathbb{E}_{\mathbb{Q}}[e^{\sigma \tilde{W}_T - \frac{\sigma^2 T}{2}}] \quad (\text{A7})$$

$$= e^{-\frac{\sigma^2 T}{2}} \frac{S_0}{\sqrt{2\pi T}} \int_{-\infty}^{\infty} dx e^{-\frac{x^2}{2T}} e^{\sigma x} \quad (\text{A8})$$

$$= S_0. \quad (\text{A9})$$

In the last step we simplified by substitution and completed the square as

$$\int_{-\infty}^{\infty} dx e^{-\frac{x^2}{2T}} e^{\sigma x} = \sqrt{T} \int_{-\infty}^{\infty} dx e^{-\frac{x^2}{2}} e^{\sigma \sqrt{T} x} \quad (\text{A10})$$

$$= \sqrt{T} e^{\frac{\sigma^2 T}{2}} \int_{-\infty}^{\infty} dx e^{-\frac{(x - \sqrt{T} \sigma)^2}{2}} \quad (\text{A11})$$

$$= \sqrt{2\pi T} e^{\frac{\sigma^2 T}{2}}. \quad (\text{A12})$$

□

Via a similar calculation we can obtain the price for the call option.

Lemma 3. *The Black-Scholes price for the European call option is given by*

$$\Pi = S_0 \Phi(d_1) - K e^{-rT} \Phi(d_2). \quad (\text{A13})$$

Proof. To compute the price we start at

$$\Pi = e^{-rT} \mathbb{E}_{\mathbb{Q}}[f(S_T)]. \quad (\text{A14})$$

Note that for the call option we can write

$$f(S_T) = \max\{0, S_T - K\} = (S_T - K) \mathbb{1}_{S_T \geq K}, \quad (\text{A15})$$

where $\mathbb{1}_x$ is the indicator function on the set x . Using the solution for S_T we have

$$f(\tilde{W}_T) = (S_0 e^{\sigma \tilde{W}_T + (r - \sigma^2/2)T} - K) \mathbb{1}_{\tilde{W}_T \geq \frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma}}. \quad (\text{A16})$$

Compute the part involving the strike price K

$$\mathbb{E} \left[\mathbb{1}_{\tilde{W}_T \geq \frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma}} \right] = \int_{\frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma\sqrt{T}}}^{\infty} dx p(x) = \Phi(d_2). \quad (\text{A17})$$

Here we use the cumulative distribution function

$$\Phi(x) = \int_{-\infty}^x dy p(y) := \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x dy e^{-\frac{y^2}{2}}. \quad (\text{A18})$$

The part involving the stock price is

$$\begin{aligned} & \mathbb{E} \left[S_0 e^{\sigma \tilde{W}_T - \sigma^2 T/2} \mathbb{1}_{\tilde{W}_T \geq \frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma}} \right] \\ &= S_0 e^{-\sigma^2 T/2} \int_{\frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma\sqrt{T}}}^{\infty} dx p(x) e^{\sigma\sqrt{T}x} \\ &= S_0 \Phi(d_1). \end{aligned} \quad (\text{A19})$$

Here,

$$d_1 = \frac{1}{\sigma\sqrt{T}} \left[\log \left(\frac{S_0}{K} \right) + \left(r + \frac{\sigma^2}{2} \right) T \right], \quad (\text{A20})$$

$$d_2 = \frac{1}{\sigma\sqrt{T}} \left[\log \left(\frac{S_0}{K} \right) + \left(r - \frac{\sigma^2}{2} \right) T \right], \quad (\text{A21})$$

This leads to the Black-Scholes price for the call option. \square

Lemma 4. *The variance of the European call option $f(S_T)$ under the risk-neutral probability measure \mathbb{Q} can be bounded as $\mathbb{V}_{\mathbb{Q}}[f(S_T)] \leq \lambda^2$ with $\lambda^2 := \mathcal{O}(\text{poly}(S_0, e^{rT}, e^{\sigma^2 T}, K))$.*

Proof. The variance is exactly computable. First note that $S_T^2 = S_0^2 e^{2\sigma \tilde{W}_T + (2r - \sigma^2)T}$. We have

$$\begin{aligned} \mathbb{E}_{\mathbb{Q}}[f(S_T)^2] &= \mathbb{E}_{\mathbb{Q}}[(S_T - K)^2 \mathbb{1}_{S_T \geq K}] \\ &= \mathbb{E}_{\mathbb{Q}}[(S_T^2 - 2S_T K + K^2) \mathbb{1}_{S_T \geq K}]. \end{aligned} \quad (\text{A22})$$

The last two terms were calculated analogously in the Black-Scholes price already. They are

$$2KE_{\mathbb{Q}}[S_T \mathbb{1}_{S_T \geq K}] = 2Ke^{rT} S_0 \Phi(d_1) \quad (\text{A23})$$

$$E_{\mathbb{Q}}[K^2 \mathbb{1}_{S_T \geq K}] = K^2 \Phi(d_2). \quad (\text{A24})$$

The first term is $\mathbb{E}_{\mathbb{Q}}[S_T^2 \mathbb{1}_{S_T \geq K}]$ which is proportional to

$$E_{\mathbb{Q}} \left[e^{2\sigma \tilde{W}_T} \mathbb{1}_{\tilde{W}_T \geq \frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma}} \right] \quad (\text{A25})$$

$$= \int_{\frac{(\log \frac{K}{S_0} - (r - \frac{\sigma^2}{2})T)}{\sigma\sqrt{T}}}^{\infty} dx p(x) e^{2\sigma\sqrt{T}x} \quad (\text{A26})$$

$$= e^{2\sigma^2 T} \Phi(d_3). \quad (\text{A27})$$

with

$$d_3 = \frac{1}{\sigma\sqrt{T}} \left[\log \left(\frac{S_0}{K} \right) + \left(r + \frac{3\sigma^2}{2} \right) T \right]. \quad (\text{A28})$$

Putting it all together we obtain

$$\begin{aligned} V_{\mathbb{Q}}[f(S_T)] &= E_{\mathbb{Q}}[f(S_T)^2] - E_{\mathbb{Q}}[f(S_T)]^2 \\ &= e^{(2r + \sigma^2)T} S_0^2 \Phi(d_3) - 2Ke^{rT} S_0 \Phi(d_1) \\ &\quad + K^2 \Phi(d_2) - (S_0 e^{rT} \Phi(d_1) - K \Phi(d_2))^2 \\ &= \mathcal{O} \left(S_0^2 e^{(2r + \sigma^2)T} + S_0 K e^{rT} + K^2 \right). \end{aligned} \quad (\text{A29})$$

We obtain an upper bound as $\mathbb{V}_{\mathbb{Q}}[f(S_T)] = \mathcal{O}(\text{poly}(S_0, e^{rT}, e^{\sigma^2 T}, K))$. \square

Appendix B: Introduction to quantum computing

Instead of the elementary bits of conventional computing, which take either the value 0 or 1, the unit of information in quantum computing is known as a *qubit*. A qubit is not restricted to being in one of the two states 0 or 1 exclusively, and can instead exist in a superposition. In quantum mechanics we use the *bra-ket* notation, with the kets $|0\rangle$ and $|1\rangle$ representing 0 and 1, respectively. The state of a qubit can then be written as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad (\text{B1})$$

in terms of so-called *amplitudes* $\alpha, \beta \in \mathbb{C}$. This means that the qubit can be in either of the states with some probability. Measuring the qubit collapses it onto $|0\rangle$ with probability $|\alpha|^2$ and onto $|1\rangle$ with probability $|\beta|^2$. Hence, $|\alpha|^2 + |\beta|^2 = 1$.

States of n qubits exist in a 2^n -dimensional Hilbert space and can be described by the vector

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle, \quad (\text{B2})$$

with $\alpha_i \in \mathbb{C}$ and $|i\rangle = |i_1\rangle \otimes |i_2\rangle \otimes \dots \otimes |i_n\rangle$ the vector of basis states over n qubits such that $[i_1, i_2, \dots, i_n]$ is the length n binary representation of i . The probability that the n qubits are in state $|i\rangle$ is equal to $|\alpha_i|^2$, where $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$. Note that the tensor product symbol \otimes denotes the joining of two quantum systems (e.g. qubits) in quantum mechanics.

Isolated quantum system evolve according to unitary transformations, i.e. so that $|\psi\rangle$ goes to $|\psi'\rangle = U|\psi\rangle$ for some unitary U (which satisfies $U^\dagger U = U U^\dagger = \mathcal{I}$, where \dagger is the conjugate transpose and \mathcal{I} is the identity operator). Unitaries can be controlled externally and also applied to collections of multiple qubits, as is often the case in this work. A standard library of one- and two-qubit unitaries can also be applied, and are often called quantum gates [32] in analogy to the gates used for binary logic operations, e.g. AND, OR, etc. This work uses the Hadamard gate \mathcal{H} , which acts on one qubit with the transformation

$$|0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad |1\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \quad (\text{B3})$$

The inverse quantum Fourier transform QFT^{-1} is also utilized in this work, which acts to perform an inverse discrete Fourier transform on the amplitudes α_j , i.e. taking the corresponding state $|j\rangle$ to

$$\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \omega^{jk} |k\rangle, \quad (\text{B4})$$

with $\omega = e^{-2\pi i/2^n}$. Unitaries acting on one or more qubits can also be controlled by another qubit, e.g. so that the unitary is enacted if the control qubit is in state $|1\rangle$ and the unitary is not enacted if the control qubit is in state $|0\rangle$. One important example is the controlled-NOT (CNOT) gate, which swaps the state of a qubit from $|0\rangle$ to $|1\rangle$ (and vice versa) whenever a control qubit is in state $|1\rangle$. Its extension is the Toffoli gate, which swaps the state of a qubit from $|0\rangle$ to $|1\rangle$ (and vice versa) only when *two* control qubits are in state $|1\rangle$.

Compositions of unitaries acting on multiple qubits, prepared in various initial states, can be created to form quantum circuits. These quantum circuits are able to carry out quantum algorithms that may perform a task faster than on a classical device, e.g. Shor's algorithm [32]. The unitary nature of quantum algorithms means that all quantum circuits are *reversible*. Reversibility has implications for the structure of quantum circuits, and marks a departure from the more familiar classical circuits. For example, the AND gate is not reversible because a single bit is returned from which the two input bits cannot be inferred in all cases.

Quantum circuits can be represented pictorially using a quantum circuit diagram, as is the case in Fig. 2 (a) and (c). Here, each qubit is represented by a horizontal line, with time moving from left to right. The unitaries are applied to various qubits by drawing a box over the qubit lines, with control from another qubit symbolized by a black dot and line connecting to the box. Measurements on the qubits are represented by the \square symbol. Finally, the CNOT gate is written as $\text{---}\bullet\text{---}\bigoplus\text{---}$, with the

upper qubit acting as the control and the lower qubit acting as the target, while the Toffoli gate is represented as $\text{---}\bullet\text{---}\bullet\text{---}\bigoplus\text{---}$, with control from the top two qubits and target on the bottom qubit.

Appendix C: Quantum circuits for arithmetic operations

In this Appendix, we review how integers and real numbers are represented and processed on a quantum computer [32]. An integer $0 \leq a < N = 2^m$ can be represented in binary with m bits x_i , $i = 0, \dots, m-1$ such that

$$a = 2^0 x_0 + 2^1 x_1 + 2^2 x_2 + \dots + 2^{m-1} x_{m-1}. \quad (\text{C1})$$

The largest number is $N-1$. The qubit encoding of the integer is where $|a\rangle$ refers to an m qubit register prepared

in the computational basis state given by the x_i , i.e. $|a\rangle = |x_0, x_1, \dots, x_m\rangle$. For real numbers $0 \leq r < 1$, we can use m bits b_i , $i = 0, \dots, m-1$ such that

$$r = \frac{b_0}{2} + \frac{b_1}{4} + \dots + \frac{b_{m-1}}{2^m} =: [.b_1, b_2, \dots, b_{m-1}]. \quad (\text{C2})$$

The accuracy is $1/2^m$. The qubit encoding of the real number is where $|r\rangle$ refers to a m qubit register prepared in the computational basis state given by the b_i , i.e. $|r\rangle = |b_0, b_1, \dots, b_{m-1}\rangle$. For signed integers or reals, we have an additional sign quantum bit $|s\rangle$.

Any operation performed on a classical computer can be written as a transformation from n to m bits, that is $F : \{0, 1\}^n \rightarrow \{0, 1\}^m$. A central result of reversible computing is that the number of input and output bits can be made the same and the function F can be mapped to a function $F' : \{0, 1\}^{n+m} \rightarrow \{0, 1\}^{n+m}$ which is given by $F'(x, y) = (x, y \oplus F(x))$. F' is a reversible function and a permutation. This permutation can be realized with a circuit that consist only of negation and Toffoli gates. If F is efficiently computable then the circuit depth is at most a polynomial in $n + m$. The classical circuit then immediately translates into a quantum circuit consisting of bit flip σ_z operations and Toffoli gates. The Toffoli gate can be broken down into a series of two qubit CNOTs and Hadamard and T gates.

We now review how basic arithmetic operations can be performed on a quantum computer. Using the following basic gates, a number of arithmetic operations can be constructed [40]:

$$\begin{array}{c} 1 \\ 2 \\ 3 \end{array} \text{---} \boxed{\text{SUM}} \text{---} = \begin{array}{c} 1 \\ 2 \\ 3 \end{array} \text{---} \begin{array}{c} \bullet \\ \bullet \\ \oplus \end{array} \text{---} \begin{array}{c} \bullet \\ \oplus \\ \oplus \end{array} \quad (\text{C3})$$

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \text{---} \boxed{\text{CY}} \text{---} = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \text{---} \begin{array}{c} \bullet \\ \bullet \\ \bullet \\ \oplus \end{array} \text{---} \begin{array}{c} \bullet \\ \oplus \\ \oplus \\ \oplus \end{array} \quad (\text{C4})$$

where CY represents the ‘carry’ operation. Composing these gates can achieve addition, multiplication, exponentiation and other operations. Numerous other works provide circuits for these operations with improved performance and lower gate requirements [41–46].

There exists a quantum circuit that performs the addition modulo N . Given two integers $0 \leq a, b < N$ we can construct a circuit of the form

$$\begin{array}{c} |a\rangle \\ |b\rangle \end{array} \text{---} \boxed{\text{ADD}} \text{---} \begin{array}{c} |a\rangle \\ |a+b\rangle \end{array}, \quad (\text{C5})$$

defined via a gate sequence of SUM and CYs in Ref. [40]. In addition, a circuit for addition modulo N can be constructed

$$\begin{array}{c} |a\rangle \\ |b\rangle \end{array} \text{---} \boxed{\text{ADD } N} \text{---} \begin{array}{c} |a\rangle \\ |a+b \bmod N\rangle \end{array}. \quad (\text{C6})$$

There also exists a quantum circuit that performs the multiplication

$$\begin{array}{c} |x\rangle \\ |0\rangle \end{array} \begin{array}{c} \boxed{MULT(a) \ N} \end{array} \begin{array}{c} |x\rangle \\ |a \times x \bmod N\rangle \end{array} \quad (C7)$$

as well as a quantum circuit that performs the exponentiation

$$\begin{array}{c} |x\rangle \\ |0\rangle \end{array} \begin{array}{c} \boxed{EXP(a)} \end{array} \begin{array}{c} |x\rangle \\ |a^x \bmod N\rangle \end{array}. \quad (C8)$$

We would like to implement the call option payoff function

$$a^+ = \max\{0, a\}. \quad (C9)$$

We can implement this as a reversible circuit

$$\begin{array}{c} |a, s\rangle \\ |0\rangle \end{array} \begin{array}{c} \boxed{MAX(0)} \end{array} \begin{array}{c} |a, s\rangle \\ |a^+\rangle \end{array}, \quad (C10)$$

which performs

$$|a, s, 0\rangle \rightarrow \begin{cases} |a, s, a\rangle & \text{if } |s\rangle = |0\rangle \\ |a, s, 0\rangle & \text{if } |s\rangle = |1\rangle \end{cases}. \quad (C11)$$

Here, the sign bit is used as a controller and a controlled addition is performed if the sign bit is positive. There exists a circuit for mapping the Brownian motion to the stock price, which consists of the basic operations shown above,

$$\begin{array}{c} |x\rangle \\ |0\rangle \end{array} \begin{array}{c} \boxed{S(\sigma, r, t)} \end{array} \begin{array}{c} |x\rangle \\ |e^{\sigma x + (r - \sigma^2/2)t}\rangle \end{array}. \quad (C12)$$

Combining the stock price with the max function, we obtain the circuit for mapping the Brownian motion outcome to the payoff for the European call option

$$\begin{array}{c} |x\rangle \\ |0\rangle \end{array} \begin{array}{c} \boxed{CALL(K, \sigma, r, T)} \end{array} \begin{array}{c} |x\rangle \\ |\tilde{v}_{\text{euro}}(x)\rangle \end{array}. \quad (C13)$$

with $\tilde{v}_{\text{euro}}(x) \equiv \tilde{v}_{\text{euro}}(x, K, \sigma, r, t)$ the bit approximation of the payoff function Eq. (36).

Appendix D: Applying the operator \mathcal{R}

This Appendix shows implementation of the operator \mathcal{R} , which rotates an ancilla based on the payoff function, where the payoff function is given by an n -bit quantum circuit, similar to circuit (C13). Using the option payoff circuit, the steps to implement \mathcal{R} are (the notation omits ancilla qubits in the $|0\rangle$ state)

$$|j\rangle \rightarrow |j\rangle |\tilde{v}(x_j)\rangle \quad (D1)$$

$$\rightarrow |j\rangle |\tilde{v}(x_j)\rangle \left(\sqrt{1 - \tilde{v}(x_j)} |0\rangle + \sqrt{\tilde{v}(x_j)} |1\rangle \right) \quad (D2)$$

$$\rightarrow |j\rangle \left(\sqrt{1 - \tilde{v}(x_j)} |0\rangle + \sqrt{\tilde{v}(x_j)} |1\rangle \right) \quad (D3)$$

$$\equiv \mathcal{R}|j\rangle |0\rangle. \quad (D4)$$

Appendix E: Preparation of the quantum state encoding the sampling distribution

Consider a probability density $p(x)$ of a single variable x . Assume we have discretized the probability over some interval, such that for some integer n , we have $\{p_j\}$ for $j = 0, \dots, 2^n - 1$. Assume that $\sum_j p_j = 1$. The task is to show an algorithm \mathcal{G} without measurements such that

$$\mathcal{G}|0^n\rangle =: |\psi\rangle = \sum_{j=0}^{2^n-1} \sqrt{p_j} |j\rangle. \quad (E1)$$

Further assume that there exists a shallow classical circuit that can efficiently compute the sums (subnorms)

$$\sum_{j=a}^b p_j \approx \int_{x_a}^{x_b} p(x) dx, \quad (E2)$$

for any $a \leq b = 0, \dots, 2^n - 1$, $a \leq b$. Thus for $m = 1, \dots, n$ we can efficiently compute the probabilities

$$p_k^{(m)} = \sum_{j=k2^{n-m}}^{(k+1)2^{n-m}-1} p_j, \quad (E3)$$

with $k = 0, \dots, 2^m - 1$.

The quantum algorithm goes as follows [33]. For $m < n$, assume we have prepared the state

$$|\psi^{(m)}\rangle = \sum_{k=0}^{2^m-1} \sqrt{p_k^{(m)}} |k\rangle. \quad (E4)$$

We would like to show by induction that we can prepare the state

$$|\psi^{(m+1)}\rangle = \sum_{k=0}^{2^{m+1}-1} \sqrt{p_k^{(m+1)}} |k\rangle. \quad (E5)$$

Define the quantities

$$f(k, m) = \frac{p_{2k}^{(m+1)}}{p_k^{(m)}}, \quad (E6)$$

where in the denominator there is the sum of all the elements of the k -th interval at the m -th discretization level and in the numerator we have the sum of the left half of these elements. This quantity allows to go up one level of discretization to $m + 1$. Also define $\theta_k^{(m)} = \arccos \sqrt{f(k, m)}$. The operation

$$|k\rangle |0\rangle \rightarrow |k\rangle |\theta_k^{(m)}\rangle \quad (E7)$$

is enabled by the efficient computability of $f(k, m)$. Now

proceed

$$|\psi^{(m)}\rangle|0\rangle|0\rangle \rightarrow \sum_{k=0}^{2^m-1} \sqrt{p_k^{(m)}}|k\rangle|\theta_k^{(m)}\rangle|0\rangle \quad (\text{E8})$$

$$\rightarrow \sum_{k=0}^{2^m-1} \sqrt{p_k^{(m)}}|k\rangle \left(\cos \theta_k^{(m)}|0\rangle + \sin \theta_k^{(m)}|1\rangle \right) \quad (\text{E9})$$

$$\equiv |\psi^{(m+1)}\rangle. \quad (\text{E10})$$

In the second step the register $|\theta_k^{(m)}\rangle$ was uncomputed.

Appendix F: Phase estimation

This Appendix shows the basic steps for phase estimation and provides an analysis of errors and the success probability.

First, the illustrative example of the single-qubit phase estimation is presented, then we review the multi-qubit setting. We follow closely references [15, 32, 47]. The single-qubit phase estimation is for demonstration purposes since it uses a known phase θ . Here, we apply the single-qubit gate

$$\mathcal{U}_z = e^{-i\frac{\theta}{2}\sigma_z} \quad (\text{F1})$$

for varying powers. In this treatment, the single qubit is effectively simulating an m -qubit register. To obtain the least significant bit of the phase, the first step is to apply $\mathcal{U}_z^{M/2}$, where M is such that $M \geq 2\pi/\epsilon$ and $M = 2^m$ for an integer m :

$$\frac{(|0\rangle + |1\rangle)}{\sqrt{2}} \rightarrow \frac{1}{\sqrt{2}} \left(e^{-i\frac{\theta}{2}\frac{M}{2}}|0\rangle + e^{+i\frac{\theta}{2}\frac{M}{2}}|1\rangle \right) \quad (\text{F2})$$

$$\rightarrow \frac{1}{2} \left(\left(e^{-i\frac{\theta}{2}\frac{M}{2}} + e^{+i\frac{\theta}{2}\frac{M}{2}} \right) |0\rangle + \left(e^{-i\frac{\theta}{2}\frac{M}{2}} - e^{+i\frac{\theta}{2}\frac{M}{2}} \right) |1\rangle \right) \quad (\text{F3})$$

$$\rightarrow \frac{1}{2} \left(2 \cos \left(\frac{\theta M}{4} \right) |0\rangle + 2i \sin \left(\frac{\theta M}{4} \right) |1\rangle \right).$$

The measurement probabilities are

$$P_0^{(m)} = \cos^2 \left(\frac{\theta M}{4} \right), \quad P_1^{(m)} = \sin^2 \left(\frac{\theta M}{4} \right). \quad (\text{F4})$$

We now use $\mathcal{U}_z^{k/2}$ for $k = m-1, \dots, 1$ to estimate the remaining bits of the phase,

$$\begin{aligned} \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) &\rightarrow \frac{1}{\sqrt{2}} \left(e^{-i\frac{\theta}{2}2^{k-1}}|0\rangle + e^{+i\frac{\theta}{2}2^{k-1}}|1\rangle \right) \\ &\rightarrow \frac{1}{\sqrt{2}} \left(e^{-i\frac{\theta}{2}2^{k-1}}|0\rangle + e^{-i\frac{\theta'}{2}2^{k-1}}e^{+i\frac{\theta}{2}2^{k-1}}|1\rangle \right). \end{aligned} \quad (\text{F5})$$

The last step applies a phase $e^{-i\theta'2^{k-1}}$ to $|1\rangle$ given by the known bits, using the identity

$$e^{-i\theta'2^{k-1}} = e^{-i\pi[b_{k+1}, \dots, b_m]}. \quad (\text{F6})$$

After another Hadamard gate, we obtain

$$\begin{aligned} &\rightarrow \frac{1}{\sqrt{2}} \left((e^{-i\frac{\theta}{2}2^{k-1}} + e^{-i\frac{\theta'}{2}2^{k-1}}e^{+i\frac{\theta}{2}2^{k-1}})|0\rangle \right. \\ &\quad \left. + (e^{-i\frac{\theta}{2}2^{k-1}} - e^{-i\frac{\theta'}{2}2^{k-1}}e^{+i\frac{\theta}{2}2^{k-1}})|1\rangle \right). \end{aligned} \quad (\text{F7})$$

From this, the measurement probabilities are given by [15]

$$\begin{aligned} P_0^{(k)} &= \frac{1}{2} + \frac{1}{2} \cos(2^k\theta - \pi[b_{k+1}, \dots, b_m]), \\ P_1^{(k)} &= 1 - P_0^{(k)}. \end{aligned} \quad (\text{F8})$$

These probabilities can be sampled to obtain an m -bit estimate of θ .

In the main text and Fig. 2, we obtain the output probability distribution for the best m -bit estimate for the phase θ using standard m -qubit phase estimation, as typically discussed in the literature [32, 47]. This presents a coherent, controlled version of the procedure above. Using the eigenstate $|\psi_\theta\rangle$ associated with the eigenvalue θ , an m -qubit register, and the operation \mathcal{Q}^c , Eq. (28), we can perform

$$\sum_{y=0}^{2^m-1} |y\rangle \mathcal{Q}^y |\psi_\theta\rangle. \quad (\text{F9})$$

In the m -qubit register, we obtain

$$\begin{aligned} &(|0\rangle + e^{i2\pi 2^{m-1}\theta}|1\rangle)(|0\rangle + e^{i2\pi 2^{m-2}\theta}|1\rangle) \times \dots \times (|0\rangle + e^{i2\pi\theta}|1\rangle) \\ &= \sum_{y=0}^{2^m-1} e^{i2\pi\theta y} |y\rangle. \end{aligned} \quad (\text{F10})$$

After applying the inverse Quantum Fourier transform we have the state

$$\frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} e^{-i2\pi\frac{xy}{2^m}} e^{i2\pi\theta y} |x\rangle. \quad (\text{F11})$$

Let $\hat{\theta}$ be the m -bit approximation of θ and $\theta = \hat{\theta} + \delta$. Using these definitions leads to

$$\begin{aligned} &\frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} e^{-i2\pi\frac{xy}{2^m}} e^{i2\pi(\hat{\theta}+\delta)y} |x\rangle = \\ &\frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} e^{i2\pi\frac{(2^m\hat{\theta}-x)y}{2^m}} e^{i2\pi\delta y} |x\rangle \end{aligned} \quad (\text{F12})$$

$$=: \sum_{x=0}^{2^m-1} \alpha_\theta(x) |x\rangle. \quad (\text{F13})$$

The estimate $\hat{\theta}$ has the amplitude

$$\alpha_\theta(2^m\hat{\theta}) = \frac{1}{2^m} \sum_{y=0}^{2^m-1} e^{i2\pi\delta y} = \frac{1}{2^m} \left(\frac{1 - e^{i2\pi\delta 2^m}}{1 - e^{i2\pi\delta}} \right), \quad (\text{F14})$$

using the geometric series. This occurs with probability

$$P(\hat{\theta}) = \frac{1}{4^m} \left| \frac{1 - e^{i2\pi\delta 2^m}}{1 - e^{i2\pi\delta}} \right|^2 = \frac{1}{4^m} \left| \frac{1 - e^{i2\pi\theta 2^m}}{1 - e^{i2\pi(\theta - \hat{\theta})}} \right|^2. \quad (\text{F15})$$

One can efficiently sample from this bit-string distribution via the individual bit probabilities given in Eqs. (F4) and (F8).

We now provide an error analysis of phase estimation. We follow closely the discussion in previous references. The phase estimation algorithm provides an estimate $\hat{\theta}$ that is accurate with ϵ , thus we have

$$|\hat{\theta} - \theta| \leq \epsilon. \quad (\text{F16})$$

The quantity of interest is the expectation value, which is related to the phase θ from Eq. (24) as

$$1 - 2\mu = \cos \frac{\theta}{2}. \quad (\text{F17})$$

We would like to determine a bound for the accuracy of this expectation value, i.e. determine

$$|\hat{\mu} - \mu|. \quad (\text{F18})$$

First, we can use the Taylor expansion $\cos((\theta \pm \epsilon)/2) = \cos \theta/2 - (\pm \epsilon) \sin(\theta/2) + \mathcal{O}(\epsilon^2)$ to arrive at the first-order bound

$$|\hat{\mu} - \mu| \leq \mathcal{O}\left(\frac{\epsilon}{2} \sin \frac{\hat{\theta}}{2}\right). \quad (\text{F19})$$

More generally, we can show the following.

Lemma 5. Assume $|\hat{\theta} - \theta| \leq \epsilon$, $0 \leq \hat{\theta} < \pi$, and $0 < \epsilon \leq 1$. Then

$$|\hat{\mu} - \mu| \leq |\cos((\hat{\theta} + \epsilon)/2) - \cos \hat{\theta}/2|. \quad (\text{F20})$$

Proof. Use the trigonometric identity for the difference between cosines

$$\begin{aligned} |\cos \hat{\theta}/2 - \cos \theta/2| &= 2|\sin((\hat{\theta} + \theta)/4) \sin((\hat{\theta} - \theta)/4)|, \\ |\cos((\hat{\theta} + \epsilon)/2) - \cos \hat{\theta}/2| &= 2|\sin((2\hat{\theta} + \epsilon)/4) \sin(\epsilon/4)|. \end{aligned}$$

Note that by definition the bit estimate $\hat{\theta} \leq \pi - \frac{\epsilon}{2}$, thus, with $|\hat{\theta} - \theta| \leq \epsilon$, we have $\frac{\hat{\theta} + \theta}{4} \leq \frac{2\hat{\theta} + \epsilon}{4} \leq \frac{\pi}{2}$. Thus

$$\left| \sin\left(\frac{\hat{\theta} + \theta}{4}\right) \right| \leq \left| \sin\left(\frac{2\hat{\theta} + \epsilon}{4}\right) \right|. \quad (\text{F21})$$

Also

$$\left| \sin\left(\frac{\hat{\theta} - \theta}{4}\right) \right| \leq \left| \sin\left(\frac{\epsilon}{4}\right) \right|. \quad (\text{F22})$$

□

We now discuss increasing the probability of success for phase estimation. The probability of observing the best m -bit approximation is lower bounded by $8/\pi^2 > 0.81$, from Eq. (F15) [32]. To boost this success probability, multiple runs of phase estimation can be performed. The median of these multiple runs will have a higher success probability [15, 22, 48], as will be shown now. Let $\hat{\theta}_1, \dots, \hat{\theta}_D$ be the results of D independent runs of phase estimation. The new estimate is the median $\hat{\theta} = \text{Median}(\hat{\theta}_1, \dots, \hat{\theta}_D)$.

Lemma 6 ([48]). Let the desired accuracy be $\epsilon > 0$. Let the probability that each sample falls outside the accuracy, i.e. $|\hat{\theta}_j - \theta| \geq \epsilon$, be $0 < \delta < 1/2$. Then the probability that the median is inaccurate is bounded by $p_f \leq \frac{1}{2} \left(2\sqrt{\delta(1-\delta)}\right)^D$.

The proof is provided in [48]. The confidence/success probability is defined as $c := 1 - p_f$. Taking the logarithm of the failure probability, $\log 2(1 - c) \leq D \log 2\sqrt{\delta(1-\delta)}$, leads to

$$|\log 1 - c| \geq D |\log 2\sqrt{\delta(1-\delta)}|, \quad (\text{F23})$$

which leads to

$$D = \mathcal{O}(|\log 1 - c|), \quad (\text{F24})$$

if $0 < \delta < 1/2$ is a constant. Hence, for a confidence of c one needs at most $\mathcal{O}(|\log 1 - c|)$ independent repetitions of phase estimation.