

Quantum Risk Analysis of Financial Derivatives

Nikitas Stamatopoulos,¹ B. David Clader,^{1,*} Stefan Woerner,² and William J. Zeng^{1,†}

¹*Goldman Sachs, New York, NY*

²*IBM Quantum, IBM Research Europe – Zurich*

We introduce two quantum algorithms to compute the Value at Risk (VaR) and Conditional Value at Risk (CVaR) of financial derivatives using quantum computers: the first by applying existing ideas from quantum risk analysis to derivative pricing, and the second based on a novel approach using Quantum Signal Processing (QSP). Previous work in the literature has shown that quantum advantage is possible in the context of individual derivative pricing and that advantage can be leveraged in a straightforward manner in the estimation of the VaR and CVaR. The algorithms we introduce in this work aim to provide an additional advantage by encoding the derivative price over multiple market scenarios in superposition and computing the desired values by applying appropriate transformations to the quantum system. We perform complexity and error analysis of both algorithms, and show that while the two algorithms have the same asymptotic scaling the QSP-based approach requires significantly fewer quantum resources for the same target accuracy. Additionally, by numerically simulating both quantum and classical VaR algorithms, we demonstrate that the quantum algorithm can extract additional advantage from a quantum computer compared to individual derivative pricing. Specifically, we show that under certain conditions VaR estimation can lower the latest published estimates of the logical clock rate required for quantum advantage in derivative pricing by up to $\sim 30x$. In light of these results, we are encouraged that our formulation of derivative pricing in the QSP framework may be further leveraged for quantum advantage in other relevant financial applications, and that quantum computers could be harnessed more efficiently by considering problems in the financial sector at a higher level.

I. INTRODUCTION

Ever since it was shown that quantum speedups were possible for Monte Carlo methods [1], various methods have been proposed to apply the potential benefits of quantum computing to the pricing of financial derivatives [2–5]. While these quantum methods allow for a quadratic speedup over state of the art classical approaches, the quantum resources required for practical advantage are significant not just in terms of circuit depth and number of qubits, but also in terms of the logical clock rate needed to match the performance of modern classical computers [6]. On the other hand, the application of quantum gradient methods utilizing the quantum oracles used for pricing [7], showed that the logical clock rate required for advantage in the task of computing the market risk of financial derivatives might be lower than that of derivative pricing itself. As such, considering the quantum oracles used in derivative pricing as components of higher-level algorithms which compute other quantities of interest of financial derivatives could lower certain quantum resources required for practical advantage.

Value at Risk (VaR) is a metric that estimates the maximum possible financial depreciation of an asset (or collection of assets) within a specific time frame at a certain confidence level [8, 9]. A related quantity, the Conditional Value at Risk (CVaR) measures the expected depreciation of an asset should losses exceed the Value at Risk, providing a risk metric for the tail of the loss distribution. Financial firms make extensive use of the VaR and CVaR metrics for their derivative holdings, aimed at assessing the financial health of their balance sheet as well as to set trading limits in order to minimize potential exposure to adverse market conditions. When individual derivative contracts are priced using Monte Carlo simulation, the computation of the VaR of portfolios of derivatives requires pricing each contract multiple times over different possible market states and then estimating the VaR from the resulting prices. For N such market states, the complexity of estimating the VaR thus scales as $\mathcal{O}(N/\epsilon^2)$, where ϵ is the accuracy with which each contract is priced. The same approach can be used to estimate the VaR by pricing of each derivative on a quantum computer and the quadratic speedup in the individual derivative pricing would carry through to the overall complexity, improving it to $\mathcal{O}(N/\epsilon)$. In Ref. [3, 10], a quantum method based on Quantum Amplitude Estimation (QAE) was introduced to estimate the VaR in cases where potential future losses of a financial asset can be modeled explicitly. While this method was shown to also provide a quadratic advantage compared to classical VaR estimation using Monte Carlo, it cannot be readily applied to cases where the financial assets in question are derivative contracts

* Current affiliation: BQP Advisors LLC

† Current affiliation: Quantonation

which are individually priced using sampling methods, and estimating the potential losses requires repeated pricing of the contracts under different market conditions.

In this article, we introduce two quantum algorithms to estimate the VaR and CVaR of derivative portfolios. The first is an extension of the QAE-based method of Ref. [3, 10]. The second is a novel algorithm based on the Quantum Signal Processing (QSP) framework [11–14], which enables nearly arbitrary polynomial transformations to quantum (sub)systems with minimal overheads. Various quantum algorithms have been formulated in this framework [15], and in most instances the QSP formulation has resulted in lower resource requirements for the algorithms [16–18], including in quantum derivative pricing [19]. Moreover, it has been shown that the quantum oracles employed in derivative pricing naturally represent block-encodings of derivative prices across different input parameters [7, 20]. Because the QSP framework relies on efficiently constructing block-encodings of quantities of interest, we hence examine how QSP can be harnessed for problems around derivative pricing.

Leveraging the QSP framework and derivative pricing oracles, we show how to apply coherent transformations to the price of a financial derivative across multiple inputs in superposition, and create tailored transformations that allow us to estimate the VaR and CVaR of derivative portfolios. We perform complexity and error analysis of the two quantum algorithms introduced and compare their performance through explicit construction of the corresponding quantum circuits and numerical simulations. Finally, we benchmark the QSP-based algorithm against classical VaR estimation methods and determine that quantum advantage is possible if suitable techniques to prepare and sample from the required probability distributions are available.

In Sec. II we formally introduce the problem of estimating the VaR and CVaR of derivatives and give an overview of the QAE-based VaR estimation method from Ref. [3, 10]. In Sec. III we generalize this method to compute the VaR of a portfolio of derivatives. In Sec. IV we give an overview of Quantum Signal Processing, show how it can be used to perform transformations to appropriately encoded derivative prices and introduce the QSP-based VaR and CVaR estimation algorithm. In Sec. VI we compare the performance of the QAE and QSP methods in terms of the quantum resources required for a target VaR accuracy, and in Sec. VII we explore the possibility of quantum advantage in estimating the VaR of derivative portfolios. We discuss our results and next steps in Sec. VIII.

II. VALUE AT RISK

The VaR computation of derivative portfolios requires modeling the possible future behavior of the market parameters underlying the derivatives. While the pricing of derivatives relies on the value of underlying parameters today (e.g. stock prices, interest rates, volatilities, etc.), which are available (or can be estimated) from public markets, computing the VaR of derivatives requires possible values of these parameters in the future. Two common modeling methods for VaR computation consist of a) using Monte Carlo simulation of the market parameters according to an appropriately defined underlying stochastic process, and b) drawing historical samples of (correlated) values for all relevant market parameters. Each such modeled market state is also called a *scenario* and can be defined as a collection of *tweaks* to today’s market parameters. To price a derivative under a scenario, the tweaks are applied to the initial market state and the derivative is priced as if the tweaked market state was today’s market. Examples of scenarios defined this way could be

1. Tweak asset X’s price up by 5%
2. Tweak asset X’s price up by 5% and asset Y’s price down by 4%
3. Tweak asset X’s price up by 5%, all points on its volatility surface up by 10% and its correlation to asset Y by 20%

In both Monte Carlo and historical VaR methods, the value of the derivative portfolio is computed under every scenario and from the resulting distribution of gains/losses, maximum levels of potential loss can be derived at different confidence levels.

More concretely, in order to compute the VaR of a derivative portfolio at a confidence level $\alpha \in [0, 1]$ (where $\alpha \geq 0.9$ is typically used), we price the portfolio under a set of N scenarios $[s_1, s_2, \dots, s_N]$ to get portfolio values $\mathcal{V} = \{V(s_i)\}$, and find the smallest portfolio value in the set such that smaller values have a probability of occurrence greater than or equal to $1 - \alpha$

$$V_\alpha = \inf \{x \mid \sum_{\{V \in \mathcal{V} \mid V \leq x\}} \mathbb{P}[V] \geq 1 - \alpha\}, \quad (1)$$

where $\mathbb{P}[V]$ denotes the probability that the portfolio has value V under the evaluated scenarios. The VaR of the portfolio at the α confidence level is then defined as $\text{VaR}_\alpha[V] \equiv V_0 - V_\alpha$, where V_0 denotes the current value of the

portfolio. This approach for computing the VaR of a portfolio can be used in both the Monte Carlo and historical methods, the only difference being the way probabilities are assigned to possible future market realizations.

The CVaR of a derivative portfolio measures the expected loss of the portfolio beyond the VaR value and can be estimated using the same set of portfolio values \mathcal{V} calculated for the VaR, by computing the quantity

$$C_\alpha = \frac{1}{1-\alpha} \sum_{\{V \in \mathcal{V} | V \leq V_\alpha\}} \mathbb{P}[V] \cdot V, \quad (2)$$

with V_α given by Eq. (1). The CVaR of the portfolio at the α confidence level is then given by $\text{CVaR}_\alpha[V] \equiv V_0 - C_\alpha$.

Classically, the VaR_α of a portfolio can be estimated by pricing the portfolio under each of the N scenarios, sorting the resulting prices from smallest to largest, and picking the value at the $1 - \alpha$ percentile as the VaR estimate. When the value of the derivative portfolio is computed classically using Monte Carlo to accuracy ϵ_p , the VaR computation requires separate Monte Carlo pricings for each scenario, and therefore the complexity of the algorithm scales as $\mathcal{O}(N/\epsilon_p^2)$.

A quantum algorithm for computing the VaR of financial assets was first presented in Ref. [3] and later refined and applied to the use case of estimating capital requirements in order to manage counterparty credit risk in Ref. [10]. The algorithm seeks to estimate the VaR of a financial asset at the α confidence level, through direct access to a unitary modeling the potential loss as a random variable L . This unitary \mathcal{L} creates a superposition

$$\mathcal{L} : |\vec{0}\rangle \rightarrow \sum_{\ell} \sqrt{p_\ell} |\ell\rangle \quad (3)$$

where the $|\ell\rangle$ register spans over possible values of L encoded in an appropriate binary representation, and p_ℓ the corresponding probability of occurrence. Picking a VaR candidate l and applying a unitary \mathcal{C} which performs the binary comparison

$$\mathcal{C} : |\ell\rangle |0\rangle \rightarrow |\ell\rangle |\ell < l\rangle, \quad (4)$$

we get the state

$$\sum_{\ell \leq l} \sqrt{p_\ell} |\ell\rangle |0\rangle + \sum_{\ell > l} \sqrt{p_\ell} |\ell\rangle |1\rangle. \quad (5)$$

The probability of measuring the last qubit in the $|0\rangle$ state gives us the probability that the loss L is smaller than l , $\mathbb{P}[|0\rangle] = \mathbb{P}[L \leq l]$. Using QAE to estimate $\mathbb{P}[|0\rangle]$ and applying a bisection search to find the smallest l_α such that $\mathbb{P}[L \leq l_\alpha] \geq 1 - \alpha$ gives us the VaR of the asset at the α confidence level, $l_\alpha = \text{VaR}_\alpha[L]$.

This quantum method is not directly applicable to the calculation of derivative VaR through the evaluation of scenario prices because in that case we do not have direct access to the unitary of Eq. (3). In the following section we show how to extend it in order to also be able to compute the VaR of derivatives by incorporating one additional application of QAE.

III. VALUE AT RISK WITH QUANTUM AMPLITUDE ESTIMATION

Quantum methods for the VaR calculation of financial derivatives rely on the existence of an oracle \mathcal{A} that encodes the price of a derivative V into an amplitude

$$\mathcal{A} : |\vec{0}\rangle \rightarrow \left(\sqrt{V} |\psi_0\rangle |0\rangle + \sqrt{1-V} |\psi_1\rangle |1\rangle \right), \quad (6)$$

for arbitrary, normalized states $|\psi_0\rangle, |\psi_1\rangle$. Different ways of constructing this oracle have been proposed in the literature [2, 4, 6]. In Appendix A, we describe in more detail the method proposed in [6] and show how it can be extended to handle the case where we want Eq. (6) to encode the value of a portfolio of derivatives instead of the price of a single derivative.

In order to compute the VaR of financial derivatives in the way described in Sec. II, we construct scenarios consisting of tweaks to (classical) market data parameters which can be encoded in some appropriate way as a computational

basis state $|s\rangle$. For example, a scenario which simultaneously tweaks d underlying parameters (also called *risk factors*), can be encoded as

$$|s\rangle = |t_1 t_2, \dots, t_d\rangle, \quad (7)$$

where t_i is a binary encoding of the tweak amount to underlying i . For such scenario encoding $|s\rangle$, we can generalize the form of operator \mathcal{A} of Eq. (6), such that it encodes the value of the portfolio under scenario s

$$\mathcal{A} : |s\rangle_q |0\rangle_n |0\rangle \rightarrow |s\rangle_q \left(\sqrt{V(s)} |\psi_0^s\rangle_n |0\rangle + \sqrt{1 - V(s)} |\psi_1^s\rangle_n |1\rangle \right), \quad (8)$$

for normalized quantum states $|\psi_0^s\rangle_n$ and $|\psi_1^s\rangle_n$. Let \mathcal{S} be an operator which loads N scenarios $[s_1, s_2, \dots, s_N]$ encoded this way in superposition, weighted by a probability denoting the likelihood of its occurrence

$$\mathcal{S} : |0\rangle_q \rightarrow \sum_{i=0}^{N-1} \sqrt{p(s_i)} |s_i\rangle_q. \quad (9)$$

The operator $\mathcal{A}_\mathcal{S} \equiv \mathcal{A}\mathcal{S}$ then encodes the portfolio value under all scenarios in superposition

$$\mathcal{A}_\mathcal{S} \equiv \mathcal{A}\mathcal{S} : |\vec{0}\rangle \rightarrow \sum_{i=0}^{N-1} \sqrt{p(s_i)} |s_i\rangle_q \left(\sqrt{V(s_i)} |\psi_0^{s_i}\rangle_n |0\rangle + \sqrt{1 - V(s_i)} |\psi_1^{s_i}\rangle_n |1\rangle \right). \quad (10)$$

Let us for now assume that each value $V(s_i)$ can be exactly represented using m qubits. The circuitry of QAE (up to measurement) can then be applied with the marked state identified as the last qubit being in the $|0\rangle$ state to get a binary representation of each $V(s_i)$ into a quantum register

$$\sum_{i=0}^{N-1} \sqrt{p(s_i)} |s_i\rangle_q |V(s_i)\rangle_m |\text{garbage}\rangle \quad (11)$$

where the state of the remaining registers $|\text{garbage}\rangle$ will depend on the choice of QAE variant used, but is not involved in the remainder of the process and can be ignored. Then the comparator of Eq. (4) can be applied to the $|V(s_i)\rangle$ register with a VaR candidate μ and an ancilla qubit initially in the $|0\rangle$ state to obtain

$$\sum_{\{i \mid V(s_i) \leq \mu\}} \sqrt{p(s_i)} |s_i\rangle_q |V(s_i)\rangle_m |0\rangle + \sum_{\{i \mid V(s_i) > \mu\}} \sqrt{p(s_i)} |s_i\rangle_q |V(s_i)\rangle_m |1\rangle. \quad (12)$$

The probability of the last qubit being in the $|0\rangle$ state gives us the probability that the portfolio value V is smaller than μ across all scenarios considered, $\mathbb{P}[|0\rangle] = \mathbb{P}[V \leq \mu]$. We can thus employ consecutive rounds of QAE to find the smallest μ_α such that $\mathbb{P}[V \leq \mu_\alpha] \geq 1 - \alpha$, which can be achieved with a bisection search over values of μ . Assuming the portfolio value today is V_0 , the VaR at the α confidence level will be given by $\text{VaR}_\alpha[V] \equiv V_0 - \mu_\alpha$. Note that if we use the *canonical* amplitude estimation [21] circuitry to generate Eq. (11), the value encoded in the resulting register will not be $V(s_i)$ but rather $\theta_i = \arcsin(\sqrt{V(s_i)})$ ¹. While quantum arithmetic can be used to compute a binary approximation to $V(s_i)$ from θ_i , since we are only interested performing the comparison $V(s_i) \leq \mu$ with $V(s_i) \in [0, 1]$, we can adjust the comparator circuit to equivalently perform $\theta_i \leq \arcsin(\sqrt{\mu})$. A diagram of the circuit required to generate Eq. (12) using canonical QAE is shown in Fig. 1.

In practice, we will not be able to represent all values $V(s_i)$ exactly in binary, and instead of Eq. (11), for each scenario value we will get a superposition over all $M = 2^m$ possible states representable using m qubits, where each state will be weighed by a probability depending on the value of $V(s_i)$

¹ Canonical QAE gives an equal superposition of two values $\theta_1 = \arcsin(\sqrt{V}) \in [0, \pi/2]$ and $\theta_2 = \pi - \theta_1 \in [\pi/2, \pi]$. Quantum arithmetic can be used to perform $\theta \rightarrow \pi - \theta$ if $\theta \in [\pi/2, \pi]$ which will ensure the register represents a value $\theta \in [0, \pi/2]$ such that $V = \sin^2(\theta)$.

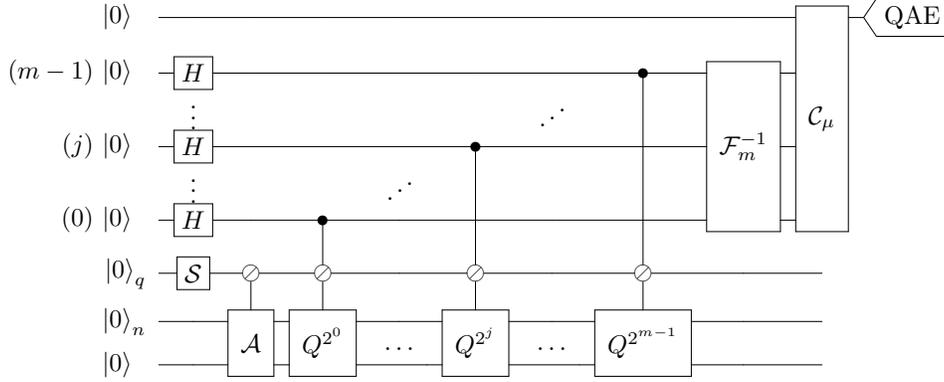


FIG. 1. Quantum circuit required to prepare the state in Eq. (12). Given a VaR candidate μ , the probability that the portfolio value is smaller than μ is encoded into the amplitude of a qubit being in the $|0\rangle$ state. First, the \mathcal{S} and \mathcal{A} operators of Eq. (9) and Eq. (8) respectively generate the superposition of all scenarios and encode the value of the portfolio under each scenario into the amplitude of the bottom qubit being in the $|0\rangle$ state. The \odot symbol indicates that an entire register acts as control to a unitary. Then, repeated controlled applications of an operator \mathcal{Q} are applied with $\mathcal{Q} = \mathcal{A}S_0\mathcal{A}^\dagger S_1$, $S_0 = \mathbb{I} - 2|\vec{0}\rangle\langle\vec{0}|$, $S_1 = \mathbb{I} - 2|1\rangle\langle 1|$ where S_0 reflects about the bottom two registers being all in the $|0\rangle$ state and S_1 reflects around the bottom qubit in the $|1\rangle$ state. After an inverse Quantum Fourier Transform, the m qubits acting as controls to the \mathcal{Q} applications contain an m -qubit approximation to the portfolio value under each scenario in superposition, as shown in Eq. (11). Finally, a comparator circuit \mathcal{C}_μ compares the binary value V encoded in the m qubits to μ and flips the top qubit if $V > \mu$, generating Eq. (12). Estimating the probability of the top qubit being in the $|0\rangle$ state using QAE and re-running this circuit for different values of μ looking for the smallest value which gives $\mathbb{P}[|0\rangle] \geq 1 - \alpha$ will give us a VaR estimate at the α confidence level.

$$\sum_{i=0}^{N-1} \sqrt{p(s_i)} |s_i\rangle_q \sum_{j=0}^{M-1} \sqrt{q_j(V(s_i))} |\tilde{V}_j\rangle_m, \quad (13)$$

with the probabilities given by [21]

$$q_j(x) = \frac{\sin^2\left(M(\tilde{V}_j - x)\pi\right)}{M^2 \sin^2\left((\tilde{V}_j - x)\pi\right)}, \quad (14)$$

which will peak around the values \tilde{V}_j closest to each $V(s_i)$. Applying the comparator unitary to perform $|\tilde{V}_j\rangle |0\rangle \rightarrow |\tilde{V}_j\rangle |\tilde{V}_j \leq \mu\rangle$ will give

$$\sum_{i=0}^{N-1} \sqrt{p(s_i)} |s_i\rangle_q \left(\sum_{\{j | \tilde{V}_j \leq \mu\}} \sqrt{q_j(V(s_i))} |\tilde{V}_j\rangle_m |0\rangle + \sum_{\{j | \tilde{V}_j > \mu\}} \sqrt{q_j(V(s_i))} |\tilde{V}_j\rangle_m |1\rangle \right), \quad (15)$$

meaning that every scenario value $V(s_i)$ will (incorrectly) contribute something to the probability of $|1\rangle$ if $V(s_i) \leq \mu$ and to $|0\rangle$ if $V(s_i) > \mu$. Increasing the number of qubits used for this *inner* application of QAE will limit the probability that is erroneously “allocated” to the wrong flag state.

Note that the QAE method used to produce Eq. (13) needs to coherently approximate $V(s_i)$, meaning highly performant iterative QAE methods [22–24] cannot be employed for this task. However, the original QAE formulation [21] as well as the QAE variant introduced in Ref. [25] can be used for coherent estimation. While the former requires the Quantum Fourier Transform (QFT) and overhead in terms of ancilla qubits, the latter can produce the approximation coherently with comparable performance to the iterative versions.

IV. VALUE AT RISK WITH QUANTUM SIGNAL PROCESSING

A. Quantum Signal Processing

Quantum Signal Processing is a technique which performs polynomial transformations to the singular values of a matrix A that has been *block-encoded* into a unitary operator². Specifically, given projectors Π and $\tilde{\Pi}$, we say that a unitary U acting on n qubits is a block-encoding of a matrix A if $A = \tilde{\Pi}U\Pi$, such that the projectors Π and $\tilde{\Pi}$ determine the location of A in U

$$U = \begin{matrix} & \Pi \\ \tilde{\Pi} & \left[\begin{array}{c} A \\ \cdot \\ \cdot \end{array} \right] \end{matrix}. \quad (16)$$

QSP consists of consecutive applications of U and U^\dagger , interleaved with projector-controlled rotation operators $\Pi_\phi = e^{i\phi(2\Pi - \mathbb{I})}$ and $\tilde{\Pi}_\phi = e^{i\phi(2\tilde{\Pi} - \mathbb{I})}$, which for phase factors $\vec{\phi} = (\phi_1, \phi_2, \dots, \phi_d)$ induce a polynomial transformation on A such that [12, 15]

$$A = \sum_k \sigma_k |w_k\rangle \langle v_k| \rightarrow P(A) = \sum_k P(\sigma_k) |w_k\rangle \langle v_k|, \quad (17)$$

where σ_k denote the singular values of A while $\{|w_k\rangle\}$, $\{|v_k\rangle\}$ are the left and right singular vectors of A respectively, and P is a d -degree polynomial. When A is Hermitian, in which case the singular values of the matrix coincide with its eigenvalues, Quantum Signal Processing effectively performs a polynomial transformation to the eigenvalues of the matrix.

More formally, define the operators

$$\mathcal{U}^{\vec{\phi}} = \begin{cases} \tilde{\Pi}_{\phi_1} U \prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U^\dagger \tilde{\Pi}_{\phi_{2k+1}} U, & \text{for odd } d, \\ \prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U^\dagger \tilde{\Pi}_{\phi_{2k}} U, & \text{for even } d \end{cases} \quad (18)$$

$$\mathcal{U}_C^{\vec{\phi}} = \left(\mathcal{U}^{\vec{\phi}} \otimes |0\rangle \langle 0| + \mathcal{U}^{-\vec{\phi}} \otimes |1\rangle \langle 1| \right). \quad (19)$$

If U is an n -qubit unitary, the polynomial transformation achieved through QSP is implemented with [15]

$$U^{\vec{\phi}} = (\mathbb{I}^{\otimes n} \otimes H) \mathcal{U}_C^{\vec{\phi}} (\mathbb{I}^{\otimes n} \otimes H) = \begin{matrix} & (\Pi \otimes |0\rangle \langle 0|) \\ (\Pi' \otimes |0\rangle \langle 0|) & \left[\begin{array}{c} P(A) \\ \cdot \\ \cdot \end{array} \right] \end{matrix} \quad (20)$$

where $\Pi' = \tilde{\Pi}$ for odd d and $\Pi' = \Pi$ for even d .

In practice, d invocations of U perform a d -degree polynomial transformation to A and the choice of phase factors $\vec{\phi} = (\phi_1, \phi_2, \dots, \phi_d)$ determine the exact polynomial that is applied. Remarkably, the reverse is also true: for $a \in [-1, 1]$ and any real polynomial $P \in \mathbb{R}(a)$, there exists a sequence of QSP phase factors $\vec{\phi} = (\phi_1, \phi_2, \dots, \phi_d)$ for which Eq. (20) holds, as long as $\deg(P) \leq d$, $|P(a)| \leq 1, \forall a \in [-1, 1]$ and P either even or odd. Additionally, for any function $f(a)$ satisfying these conditions we can first identify a polynomial approximation to the function, determine the QSP phase factors $\vec{\phi}$ for that polynomial, and use QSP to approximately apply $f(a)$ to the singular values of the block-encoded matrix. Various methods have been proposed to generate polynomial approximations to generic functions, where the approximation is either constructed analytically [15, 26, 27] or with optimization-based numerical methods [28, 29].

² This technique was originally introduced as Quantum Singular Value Transformation in Ref. [12], but here we use the nomenclature of Ref. [15], which refers to the underlying method as Quantum Signal Processing, encompassing a variety of applications.

B. Algorithm for Value at Risk

Defining $\tilde{\Pi} \equiv I^{\otimes(n+q)} \otimes |0\rangle\langle 0|$ and $\Pi \equiv I^{\otimes q} \otimes (|0\rangle\langle 0|)^{\otimes(n+1)}$, observe that $\tilde{\Pi}\mathcal{A}\Pi$ with \mathcal{A} given by Eq. (8), is a rank-1 matrix with a single non-trivial singular value $\sqrt{V(s)}$ [12]. Therefore, we can use QSP with the operator \mathcal{A} acting as the block encoding unitary of Eq. (16) to apply polynomial transformations to the values $\sqrt{V(s)}$ as discussed in the previous section. After the superposition of scenarios is created with the unitary \mathcal{S} of Eq. (9), the unitary $U^{\vec{\phi}}$ of Eq. (20) based on $\mathcal{U}^{\vec{\phi}}$ with $U = \mathcal{A}$ produces the state

$$U^{\vec{\phi}}\mathcal{S}|0\rangle_q|0\rangle_{n+1}|0\rangle = \sum_{i=0}^{N-1} \sqrt{p(s_i)}|s_i\rangle_q \begin{cases} \left(P(\sqrt{V(s_i)})|\psi_0\rangle_n|0\rangle_2 + \sqrt{1-P(\sqrt{V(s_i)})^2}|\psi_1\rangle_n|0_\perp\rangle_2 \right), & \text{for odd } d, \\ \left(P(\sqrt{V(s_i)})|0\rangle_{n+2} + \sqrt{1-P(\sqrt{V(s_i)})^2}|0_\perp\rangle_{n+2} \right), & \text{for even } d, \end{cases} \quad (21)$$

for normalized quantum states $|\psi_0\rangle_n$, $|\psi_1\rangle_n$, and $|0_\perp\rangle_a$ denoting a normalized state orthogonal to $|0\rangle_a$. The polynomial $P(x)$ is determined by the choice of phase factors $\vec{\phi}$.

If we can find phase factors such that $P(x)$ is an approximation to the threshold function

$$\theta_\mu(x) = \begin{cases} 1, & x \leq \mu \\ 0, & x > \mu, \end{cases} \quad (22)$$

Eq. (21) will give us

$$\sum_{\{i \mid \sqrt{V(s_i)} \leq \mu\}} \sqrt{p(s_i)}|G\rangle + \sum_{\{i \mid \sqrt{V(s_i)} > \mu\}} \sqrt{p(s_i)}|G_\perp\rangle, \quad (23)$$

where $|G\rangle = |0\rangle_2$ for odd d and $|G\rangle = |0\rangle_{n+2}$ for even d , ignoring the (normalized) state of the remaining registers for clarity. The probability of measuring $|G\rangle$ will then be equal to the probability that the portfolio value across all scenarios is smaller than μ^2

$$\mathbb{P}(|G\rangle) = \sum_{\{s \mid V(s) \leq \mu^2\}} p(s). \quad (24)$$

Therefore, the smallest value of $\mu = \mu_\alpha$ for which the probability in Eq. (24) satisfies $\mathbb{P}(|G\rangle) \geq 1 - \alpha$ gives us the value of $V_\alpha = \mu_\alpha^2$ we are looking for in Eq. (1) in order to estimate the VaR of the portfolio at the α confidence level. We can find such μ_α by performing bisection search over values of $\mu \in [0, 1]$ and applying QAE to estimate the probability in Eq. (24) in each iteration. The bisection search ends when we find a value of μ for which our estimate of $\mathbb{P}(|G\rangle)$ is ϵ -close to $1 - \alpha$ for some value of ϵ determined by the target accuracy of the QAE method employed. By the union bound, if each round of QAE is performed at confidence level $1 - \alpha_A$, the confidence of our final VaR estimate after k rounds will be $1 - k \cdot \alpha_A$.

This approach can be optimized by observing that not every bisection step needs to use the same accuracy ϵ or confidence level parameter α_A for QAE. Because at every step we need to estimate whether the value $\mathbb{P}(|G\rangle)$ in Eq. (24) is ϵ -close to $1 - \alpha$, we can employ QAE with accuracy $\epsilon_k > \epsilon$ in cases where $\mathbb{P}(|G\rangle)$ is significantly far from $1 - \alpha$. Using iterative amplitude estimation routines such as IQAE [22], we can adaptively increase the accuracy of the estimate by taking progressively more samples until we can determine whether $|\mathbb{P}(|G\rangle) - (1 - \alpha)| \leq \epsilon$ with some confidence $1 - \alpha_A$ and then decide on the next bisection search step accordingly. This observation is illustrated in Fig. 2.

The confidence level $1 - \alpha_A$ of QAE can similarly be adjusted in each round. If the target confidence level of the VaR estimation is denoted as CL, by the union bound the confidence level α_k in the k -th QAE invocation can be adjusted as long as $\text{CL} = 1 - \sum_k \alpha_k$ is satisfied. Therefore, the total probability of failure $1 - \text{CL}$ can be distributed across each QAE round as desired. One possible choice is to allocate higher probability of failure when the QAE target error ϵ_k is small, which would decrease the constant factor of the $\mathcal{O}(1/\epsilon_k)$ complexity of QAE when $1/\epsilon_k$ is large. The pseudocode for this QSP-based algorithm is given in Algorithm 1 and the circuit required to generate Eq. (24) is shown in Fig. 3.

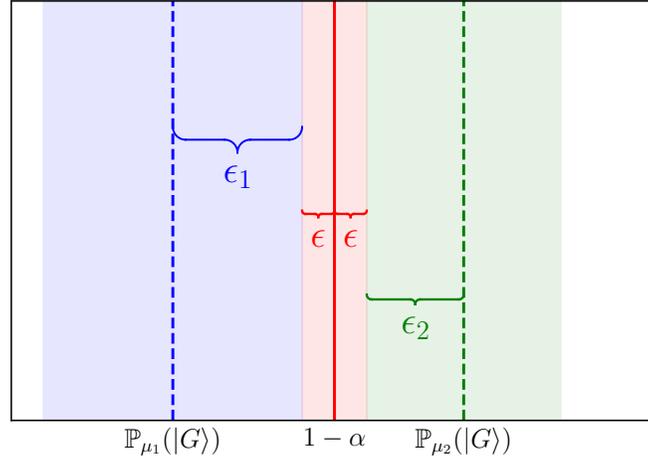


FIG. 2. For each round of bisection search in Algorithm 1, we are trying to determine whether for a given value μ , the value $\mathbb{P}(|G\rangle)$ of Eq. (24) is ϵ -close to $1 - \alpha$. However, the accuracy with which we estimate the value $\mathbb{P}(|G\rangle)$ can be relaxed depending on how far $\mathbb{P}(|G\rangle)$ is from $1 - \alpha$. For example, for the values $\mathbb{P}_{\mu_1}(|G\rangle)$ and $\mathbb{P}_{\mu_2}(|G\rangle)$ corresponding to some thresholds μ_1, μ_2 shown above, each probability only needs to be estimated with accuracy $\epsilon_1, \epsilon_2 > \epsilon$ respectively in order to proceed with the bisection search.

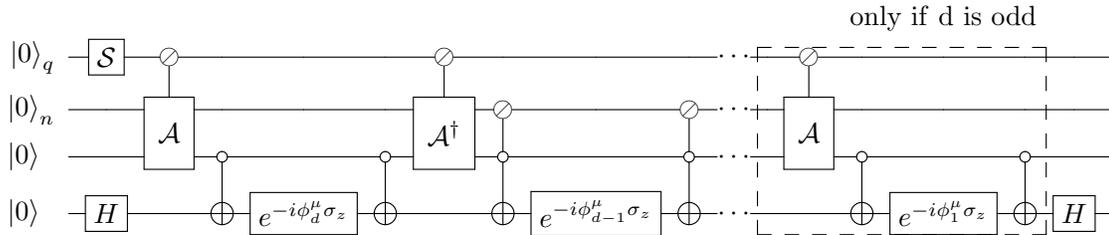


FIG. 3. Circuit diagram depicting one iteration of the QSP-based VaR estimation method described algorithm in Sec. IV B and outlined in Algorithm 1. Given a VaR candidate μ , the probability that the portfolio value is smaller than μ^2 is encoded into the amplitude of a marked state. The \mathcal{S} operator of Eq. (9) is first applied to generate the superposition of all scenarios considered. Then, a d -degree polynomial approximation to the threshold function $\theta_\mu(x)$ of Eq. (22) along with the corresponding phase factors $\vec{\phi}^\mu = (\phi_1^\mu, \phi_2^\mu, \dots, \phi_d^\mu)$ are computed, and the generated polynomial transformation is implemented using the QSP unitary of Eq. (21), which consists of alternating applications of controlled phase rotations and the block-encoding operator \mathcal{A} of Eq. (8). After the transformation, the probability of measuring $|G\rangle = |0\rangle_2$ in the bottom two qubits if d is odd or $|G\rangle = |0\rangle_{n+2}$ in the bottom $n + 2$ qubits if d is even, gives us the probability that the portfolio value across all scenarios is smaller than μ^2 as shown in Eq. (24). Estimating this probability using QAE and re-running this circuit for different values of μ looking for the smallest value which gives $\mathbb{P}[|G\rangle] \geq 1 - \alpha$ will give us a VaR estimate at the α confidence level.

C. Conditional Value at Risk

Once we have estimated the VaR, the CVaR of derivatives can be also be estimated using QSP by applying a modified threshold function which is linear below the threshold and zero above

$$\tilde{\theta}_\mu(x) = \begin{cases} x, & x \leq \mu \\ 0, & x > \mu, \end{cases} \quad (25)$$

at $\mu = \mu_\alpha$ where μ_α is the threshold estimated in the VaR calculation such that the probability in Eq. (24) equals $1 - \alpha$. Using QSP to apply the transformation of Eq. (21) with $P(x)$ now a polynomial approximation to this function gives us

Algorithm 1: VaR Estimation using Quantum Signal Processing

Input: Amplitude Estimation accuracy ϵ_A

Input: VaR confidence level α

1 Set $\mu_l = 0$ and $\mu_h = 1$

2 Set $k = 0$

3 **while** *True* :

4 Apply the S operator of Eq. (9) to generate the superposition of all scenarios used to compute the VaR of a derivative portfolio whose value today is V_0

$$\sum_{i=0}^{N-1} \sqrt{p(s_i)} |s_i\rangle$$

5 Pick $\mu = \mu_l + (\mu_h - \mu_l)/2$

6 Generate a d -degree polynomial and corresponding phase factors $\vec{\phi}^\mu = (\phi_1^\mu, \phi_2^\mu, \dots, \phi_d^\mu)$ such that the polynomial transformation $P(A)$ in Eq. (20) is an approximation to the threshold function $\theta_\mu(x)$ of Eq. (22).

7 Apply the generated polynomial transformation using QSP and the block-encoding operator \mathcal{A} of Eq. (8) such that the probability of the last qubits being in the $|G\rangle$ state with $|G\rangle = |0\rangle_2$ for odd d and $|G\rangle = |0\rangle_{n+2}$ for even d becomes

$$\mathbb{P}(|G\rangle) = \sum_{\{s \mid V(s) \leq \mu^2\}} p(s)$$

8 Iteratively estimate the probability $\mathbb{P}(|G\rangle)$ using QAE with decreasing estimation error $\epsilon_k \geq \epsilon_A$ to determine bounds satisfying $\mathbb{P}(|G\rangle) \in [p_l, p_h]$ with $p_h - p_l \leq 2\epsilon_k$ at confidence level $1 - \alpha_k$, until we either determine that $1 - \alpha \notin [p_l, p_h]$ or the target estimation error reaches $\epsilon_k = \epsilon_A$.

9 Set $k = k + 1$

10 **if** $p_h < 1 - \alpha$:

11 Set $\mu_l = \mu$

12 **elif** $p_l > 1 - \alpha$:

13 Set $\mu_h = \mu$

14 **else:**

15 Return VaR_α estimate $V_0 - \mu^2$ with confidence $1 - \sum_k \alpha_k$

$$\sum_{\{i \mid \sqrt{V(s_i)} \leq \mu_\alpha\}} \sqrt{p(s_i)} \sqrt{V(s_i)} |G\rangle + \sum_{\{i \mid \sqrt{V(s_i)} \leq \mu_\alpha\}} \sqrt{p(s_i)} \sqrt{1 - V(s_i)} |G_\perp\rangle + \sum_{\{i \mid \sqrt{V(s_i)} > \mu\}} \sqrt{p(s_i)} |G_\perp\rangle. \quad (26)$$

Applying QAE to estimate the probability of measuring $|G\rangle$ will give us the expected value of the portfolio below the VaR level

$$\mathbb{P}(|G\rangle) = \sum_{\{s \mid V(s) \leq \mu_\alpha^2\}} p(s) V(s). \quad (27)$$

Dividing this quantity by $\mathbb{P}[V \leq \mu_\alpha^2]$ will give us C_α of Eq. (2) and the CVaR of the portfolio is then given by $V_0 - C_\alpha$, if the portfolio value today is V_0 .

Because the threshold function of Eq. (25) is discontinuous at $x = \mu$, we will need a high degree polynomial to generate a close approximation to the function. We can however avoid the discontinuity at $x = \mu$ by inverting the threshold function in Eq. (25) in the region below μ such that

$$\hat{\theta}_\mu(x) = \begin{cases} \mu - x, & x \leq \mu \\ 0, & x > \mu. \end{cases} \quad (28)$$

Applying this transformation instead at $\mu = \mu_\alpha$ gives us

$$\mathbb{P}(|G\rangle) = \sum_{\{s \mid V(s) \leq \mu_\alpha^2\}} p(s) (\mu_\alpha - V(s)), \quad (29)$$

from which we compute

$$\hat{C}_\alpha = \frac{1}{\mathbb{P}[V \leq \mu_\alpha^2]} \sum_{\{s \mid V(s) \leq \mu_\alpha^2\}} p(s) (\mu_\alpha - V(s)) = \left(\frac{1}{\mathbb{P}[V \leq \mu_\alpha^2]} \sum_{\{s \mid V(s) \leq \mu_\alpha^2\}} p(s) \mu_\alpha \right) - C_\alpha = \mu_\alpha - C_\alpha. \quad (30)$$

The CVaR can then be calculated as $\text{CVaR}_\alpha[V] = V_0 - C_\alpha = V_0 - (\mu_\alpha - \hat{C}_\alpha)$.

D. Threshold Function Approximation using QSP

In order to apply the threshold functions in Eq. (22) and Eq. (25) using QSP, we first need to find polynomials which approximate them. The polynomial transformations possible through the QSP framework must have definite parity (even or odd), and because we aim to transform real positive amplitudes representing derivative prices, we restrict our attention to the interval $[0, 1]$ due to symmetry. For the threshold function in Eq. (22) used for the VaR calculation, we follow the method in Ref. [29] and look for a real polynomial $f(x)$ satisfying

$$|f(x) - c| \leq \epsilon, \quad \forall x \in [0, \mu - \Delta/2]; \quad |f(x)| \leq \epsilon, \quad \forall x \in [\mu + \Delta/2, 1] \quad (31)$$

where c is chosen close to 1 but preferably slightly smaller to avoid overshooting, ϵ controls how far away the function is allowed to deviate from the values 0 and 1, and Δ is a gap parameter which controls the steepness of the jump from 1 to 0 at μ where $\theta_\mu(x)$ is discontinuous. Smaller values of Δ will create a better approximation to the function in the interval $[\mu - \Delta/2, \mu + \Delta/2]$ and a smaller value of ϵ will correspondingly allow for a better approximation in the interval $[0, \mu - \Delta/2] \cup [\mu + \Delta/2, 1]$.

While there are analytical methods of obtaining an approximation $f(x)$ to this function with a polynomial of degree $\mathcal{O}((1/\Delta) \log(1/\epsilon))$ [30], we employ the optimization-based method described in Ref. [29] which generates near-optimal approximations without relying on any analytic computation. This method constructs a linear combination of even Chebyshev polynomials with some unknown coefficients $\{c_k\}$

$$f(x) = \sum_{k=0}^{d/2} c_k T_{2k}(x), \quad (32)$$

and aims to find the coefficients $\{c_k\}$ which give the best approximation to $\theta_\mu(x)$. This process is formulated as a discrete optimization problem by discretizing the interval $[0, 1]$ using M grid points generated by the roots of Chebyshev polynomials $\left\{x_j = -\cos \frac{j\pi}{M-1}\right\}_{j=0}^{M-1}$ and solving the following minimax optimization problem for the coefficients $\{c_k\}$

$$\begin{aligned} \min_{\{c_k\}} \max \left\{ \max_{x_j \in [0, \mu - \Delta/2]} |f(x_j) - c|, \max_{x_j \in [\mu + \Delta/2, 1]} |f(x_j)| \right\} \\ \text{s.t. } f(x_j) = \sum_k c_k A_{jk}, \quad |f(x_j)| \leq c, \quad \text{for } j = 0, 1, \dots, M-1, \end{aligned} \quad (33)$$

where A_{jk} is a matrix of coefficients defined as $A_{jk} = T_{2k}(x_j)$ for $k = 0, 1, \dots, d/2$.

We follow the same procedure to construct a polynomial approximation to the inverted threshold function $\hat{\theta}_\mu$ of Eq. (28) used in the estimation of CVaR. In this case, we solve the minimax optimization problem

$$\min_{\{c_k\}} \max \left\{ \max_{x_j \in [0, \mu - \Delta/2]} |f(x_j) - (\mu - x_j)|, \max_{x_j \in [\mu + \Delta/2, 1]} |f(x_j)| \right\}, \quad (34)$$

subject to the same constraints as Eq. (33). The term $|f(x_j) - (\mu - x_j)|$ now enforces that the polynomial approximation is close to $\mu - x$ in the interval $[0, \mu - \Delta/2]$. Polynomial approximations generated using the optimization-based method described in this section for the threshold functions $\theta_\mu(x)$ and $\hat{\theta}_\mu(x)$ at $\mu = 0.5$ are shown in Fig. 4.

While the calculation of CVaR is just as (or more) important than VaR as a business use case, in practice they are both formulated similarly in the QSP framework. In fact, computing the CVaR using QSP requires first the computation of VaR and the correct threshold to be identified. As such, in the following sections we focus on the performance and estimation errors of the VaR algorithm which is the core component in the estimation of both risk metrics.

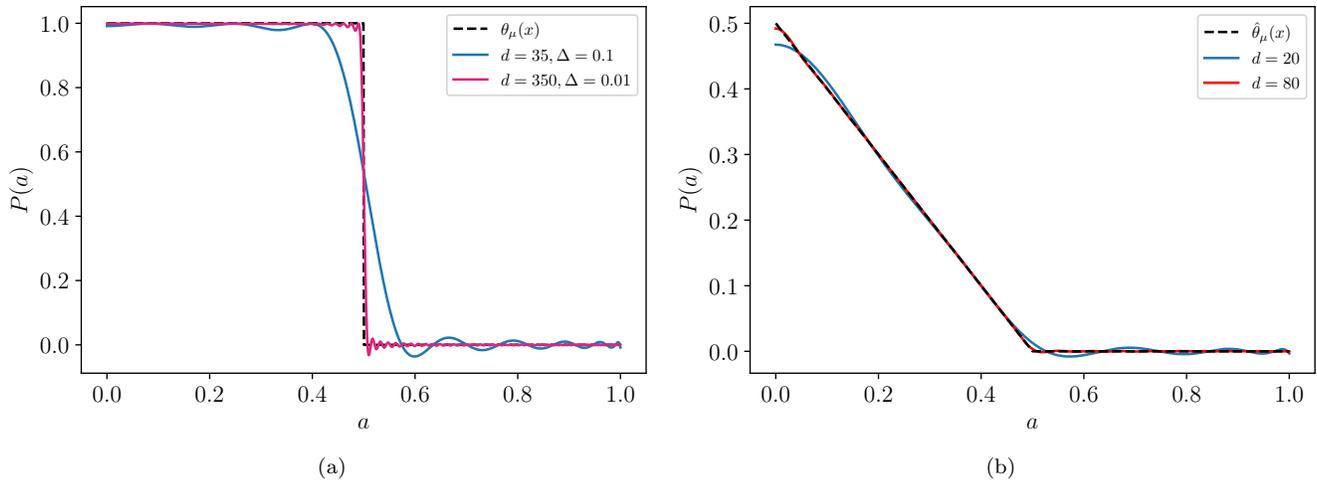


FIG. 4. Polynomial approximations generated using the optimization-based method described in Sec. IV D. Using Quantum Signal Processing we can perform the transformation $a \rightarrow P(a)$ for each singular value a of a block-encoded unitary, where increasing the degree of the polynomial $P(a)$ will result in a better approximation to the target function. (a) Approximations to the threshold function of Eq. (22) at $\mu = 0.5$ used for the estimation of VaR, generated by optimizing Eq. (33) for different polynomial degrees d and gap parameter Δ , with $\epsilon = 1 - c = 10^{-3}$. (b) Approximations to the linear threshold function of Eq. (28) used for the estimation of CVaR, generated using Eq. (34) for different polynomial degrees d , $\Delta = 10^{-3}$ and $\epsilon = 1 - c = 10^{-3}$. Avoiding the discontinuity at $a = \mu$ by using the function of Eq. (28) instead of Eq. (25) in the calculation of CVaR results in significantly lower degree polynomials required to approximate the function with high degree of accuracy.

V. COMPLEXITY AND ERROR ANALYSIS

The VaR calculation methods discussed so far rely on estimating the probability that a portfolio value X will be less than some threshold μ . In the continuous setting, this can be written as

$$P[X \leq \mu] = \int_0^1 \theta_\mu(x) p(x) dx, \quad (35)$$

where $p(x)$ denotes the probability of value x occurring, $\theta_\mu(x)$ is the threshold function defined in Eq. (22) and the portfolio values have been normalized to lie in $[0, 1]$. The estimation of VaR is then the solution to the inverse problem, where given a confidence level α , we try to determine the value μ_α giving $P[X \leq \mu_\alpha] \geq 1 - \alpha$, which can be formulated as finding the root of

$$\tilde{f}(\mu_\alpha) = (1 - \alpha) - P[X \leq \mu_\alpha]. \quad (36)$$

The error in the estimation of VaR will be determined by the accuracy with which we can approximate Eq. (35) and the accuracy in finding the root of Eq. (36), given the approximation to $P[X \leq \mu_\alpha]$. An error δ in the estimate of $P[X \leq \mu_\alpha]$ will result in error ϵ in the estimate of the VaR given by

$$\epsilon = \delta \cdot \left. \frac{dF^{-1}(p)}{dp} \right|_{p=1-\alpha} + \mathcal{O}(\delta^2), \quad (37)$$

where $F^{-1}(p)$ is the inverse cumulative distribution function of the distribution $p(x)$, assuming $F^{-1}(p)$ is analytic at $p = 1 - \alpha$. We now examine the accuracy with which $P[X \leq \mu_\alpha]$ can be estimated by various methods, which to first order gives the accuracy of the VaR estimate (with a constant factor depending on the shape of the probability distribution $p(x)$ at μ_α).

A. Classical and Semi-Classical Methods

Classical methods for the VaR estimation of financial derivatives use two nested Monte Carlo invocations; an inner round to estimate the price of the derivative x (see Eq. (A1)), and an outer round sampling scenarios to estimate the integral of Eq. (35). When N scenarios are used to approximate the integral and the derivative price for each scenario is evaluated to accuracy ϵ_p , the approximation P' can be written as

$$P'[X \leq \mu] = \frac{1}{N} \sum_{i=1}^N \theta_\mu(x_i + \epsilon_p). \quad (38)$$

Normalizing the prices x_i to lie in the interval $[0, 1]$, the error of this approximation is

$$|P' - P| = \left| \frac{1}{N} \sum_{i=1}^N \theta_\mu(x_i + \epsilon_p) - \int_0^1 \theta_\mu(x) p(x) dx \right| \leq \epsilon_\theta + \epsilon_S$$

with $\epsilon_\theta \equiv \left| \frac{1}{N} \sum_{i=1}^N \theta_\mu(x_i + \epsilon_p) - \frac{1}{N} \sum_{i=1}^N \theta_\mu(x_i) \right|$, $\epsilon_S \equiv \left| \frac{1}{N} \sum_{i=1}^N \theta_\mu(x_i) - \int_0^1 \theta_\mu(x) p(x) dx \right|$, (39)

where ϵ_θ is the error stemming from the evaluation of the threshold function using an approximate derivative price, and ϵ_S the error from approximating the integral with Monte Carlo sampling. We can derive the asymptotic dependence of ϵ_θ on ϵ_p by considering the expression for ϵ_θ in Eq. (39) in the continuous limit

$$\begin{aligned} \epsilon_\theta &= \left| \int_0^1 \theta_\mu(x + \epsilon_p) p(x) dx - \int_0^1 \theta_\mu(x) p(x) dx \right| \\ &= \left| \int_0^\mu p(x) dx + \int_\mu^{\mu+\epsilon_p} p(x) dx - \int_0^\mu p(x) dx \right| = \left| \int_\mu^{\mu+\epsilon_p} p(x) dx \right| \approx p(\mu) \epsilon_p + O(\epsilon_p^2), \end{aligned} \quad (40)$$

where the last approximation follows by considering the left Riemann sum approximation to the integral³. It then follows that $\epsilon_\theta = \mathcal{O}(\epsilon_p)$ with the constant factor depending on the shape of the probability distribution $p(x)$ at μ . Due to the two nested Monte Carlo evaluations, the complexity of this method scales as $\mathcal{O}(\epsilon_S^{-2} \epsilon_p^{-2})$.

The *semi-classical* approach proceeds similarly as in the purely classical case, the only difference being that the derivative price for each scenario is evaluated on a quantum computer using amplitude estimation. In this case, the approximation error is the same, but the quadratic speedup of amplitude estimation makes the approximation to $P[X \leq \mu]$ scale as $\mathcal{O}(\epsilon_S^{-2} \epsilon_p^{-1})$.

B. QAE VaR

With the QAE VaR method we create a superposition over scenarios $\sum_i \sqrt{p(s_i)} |s_i\rangle$ with the operator of Eq. (9), evaluate the derivative price x_i for each scenario s_i in superposition using amplitude estimation, and after applying a binary comparator at μ , we read out the approximate probability $P'[X \leq \mu]$ using another round of QAE (see Fig. 1). Contrary to the classical and semi-classical methods, in this case we are not limited to classical sampling of scenarios if we have a way to load the superposition of scenarios using a closed-form expression for $p(s_i)$. As such, the approximation $P'[X \leq \mu]$ in this case can be written as

$$P'[X \leq \mu] = \sum_{i=1}^N \theta_\mu(x_i + \epsilon_p) p(x_i) \delta x + \epsilon_A, \quad (41)$$

³ We can also consider the pricing error contribution to the threshold function in Eq. (40) as $\theta_\mu(x - \epsilon_p)$ and the same asymptotic dependence can be derived by considering the right Riemann sum approximation to the integral.

where ϵ_p is the approximation error of using amplitude estimation to approximate the derivative price x_i and ϵ_A is the error from the outer amplitude estimation. The the sum over x_i can be a (multi-dimensional) Riemann sum approximating an integral if $p(x)$ can directly loaded into the superposition of scenarios. If the scenarios are instead sampled from $p(x)$, the above expression becomes a Monte Carlo approximation with $p(x_i)\delta x = 1/N$. In this case the error of the approximation can be written as

$$|P' - P| = \left| \sum_{i=1}^N \theta_\mu(x_i + \epsilon_p)p(x_i)\delta x - \int_0^1 \theta_\mu(x)p(x)dx \right| + \epsilon_A \leq \epsilon_\theta + \epsilon_S + \epsilon_A$$

$$\text{where } \epsilon_\theta \equiv \left| \sum_{i=1}^N \theta_\mu(x_i + \epsilon_p)p(x_i)\delta x - \sum_{i=1}^N \theta_\mu(x_i)p(x_i)\delta x \right|, \quad \epsilon_S \equiv \left| \sum_{i=1}^N \theta_\mu(x_i)p(x_i)\delta x - \int_0^1 \theta_\mu(x)p(x)dx \right|, \quad (42)$$

and the asymptotic dependence of ϵ_θ on ϵ_p follows similarly to the classical case, with $\epsilon_\theta = \mathcal{O}(\epsilon_p)$.

If the cost of loading the scenario superposition is $C(\epsilon_S)$, the total complexity of estimating $P[X \leq \mu]$ using QAE VaR is $\mathcal{O}((C(\epsilon_S) + \epsilon_p^{-1})\epsilon_A^{-1})$. When the individual scenarios in the superposition are selected by sampling an appropriate probability distribution similarly to the classical and semi-classical approach, a sampling error ϵ_S requires $\epsilon_S = \mathcal{O}(1/\sqrt{N})$ scenarios. As this superposition can be loaded in $\mathcal{O}(\log N)$ depth⁴ [31], the total complexity becomes $\mathcal{O}((\log(1/\epsilon_S^2) + \epsilon_p^{-1})\epsilon_A^{-1}) = \tilde{\mathcal{O}}(\epsilon_p^{-1}\epsilon_A^{-1})$, where the tilde notation ignores poly-logarithmic terms.

C. QSP VaR

The QSP VaR algorithm follows similar steps as the QAE variant, but in this case the derivative prices are computed with the accuracy allowed of the \mathcal{A} operator from Eq. (6) and do not incur an approximation error from amplitude estimation. While there is still approximation error in the implementation of \mathcal{A} , it can be made exponentially small by increasing the number of qubits in the construction of the operator with only logarithmically increasing the oracle complexity of the algorithm [6] and as such we ignore it in this error analysis. On the other hand, because in this case we use QSP to approximate the threshold function of Eq. (22), we do not implement exactly the threshold function $\theta_\mu(x)$ but rather an approximation, which we denote as $\theta'_\mu(x)$. As in the case of QAE VaR, the preparation of the scenarios is not limited to sampling if there is a method to load the scenario superposition efficiently, so we use the same notation as the previous section to write the approximation to $P[X \leq \mu]$ in this case as

$$P'[X \leq \mu] = \sum_{i=1}^N \theta'_\mu(x_i)p(x_i)\delta x + \epsilon_A. \quad (43)$$

The approximation error is then given by

$$|P' - P| = \left| \sum_{i=1}^N \theta'_\mu(x_i)p(x_i)\delta x - \int_0^1 \theta_\mu(x)p(x)dx \right| + \epsilon_A \leq \epsilon_\theta + \epsilon_S + \epsilon_A$$

$$\text{where } \epsilon_\theta \equiv \left| \sum_{i=1}^N \theta'_\mu(x_i)p(x_i)\delta x - \sum_{i=1}^N \theta_\mu(x_i)p(x_i)\delta x \right|, \quad \epsilon_S \equiv \left| \sum_{i=1}^N \theta_\mu(x_i)p(x_i)\delta x - \int_0^1 \theta_\mu(x)p(x)dx \right|. \quad (44)$$

Contrary to the classical, semi-classical and QAE cases, QSP allows us to bound the error from the first term ϵ_θ . For a choice of the gap parameter Δ , picking a polynomial degree of $d = \mathcal{O}(1/\Delta)$ will guarantee that $\epsilon_\theta \leq \mathcal{O}(\Delta) \sim \mathcal{O}(1/d)$ [29, 30]. In Fig. 5 we show numerically how this approximation error scales with the degree of the polynomial used to approximate $\theta'_\mu(x)$ for $\Delta = 10^{-3}$ when the polynomial is generated using the method described in Sec. IV D. The

⁴ It is important to highlight that while this superposition can generally be loaded in $\mathcal{O}(\log N)$ depth, in practice, this complexity might include very large constant factors, as discussed in [31].

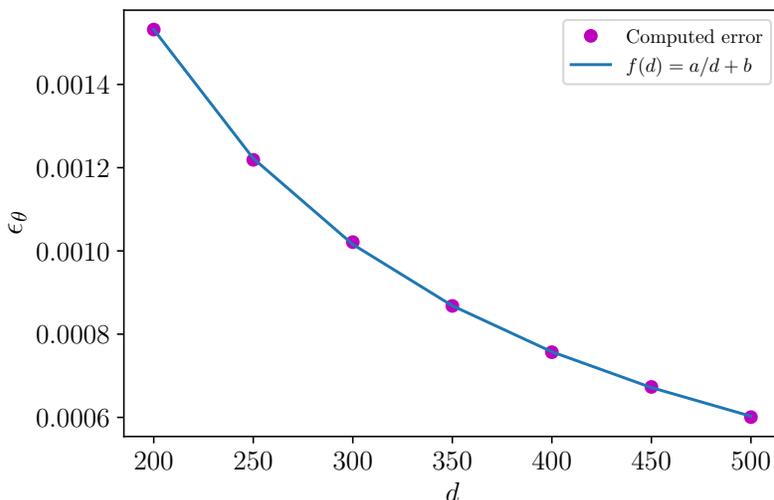


FIG. 5. The approximation error ϵ_θ in the estimation of $P[X \leq \mu]$ due to the polynomial approximation used in the QSP VaR method as shown in Eq. (44). We pick the gap parameter $\Delta = 10^{-3}$ and numerically compute the approximation error of d -degree polynomials generated using the optimization-based method described in Sec. IV D. This error is expected to scale as $\mathcal{O}(1/d)$ and fitting a function of the form $f(d) = a/d + b$ to the data in this case gives us $a \approx 0.31$.

complexity of estimating $P[X \leq \mu]$ then scales as $\mathcal{O}((C(\epsilon_S) + \epsilon_\theta^{-1})\epsilon_A^{-1})$, where again $C(\epsilon_S)$ denotes the cost of loading the superposition over scenarios. In the case where the scenarios are selected by sampling a distribution and loaded in superposition in $\mathcal{O}(\log(N))$ depth, we get $\tilde{\mathcal{O}}(\epsilon_\theta^{-1}\epsilon_A^{-1})$.

From the analysis in this section, we see that for a target error ϵ in the estimation of the probability in Eq. (35), classical nested Monte Carlo scales as $\mathcal{O}(\epsilon^{-4})$ and the semi-classical method scales as $\mathcal{O}(\epsilon^{-3})$, where the advantage stems from the quadratic advantage of QAE compared to classical sampling in the inner (pricing) Monte Carlo step. The semi-classical method cannot provide further advantage in the outer Monte Carlo sampling step, as both of these methods are limited to sampling scenarios from an appropriate probability distribution. When also restricted to scenario sampling, the QAE and QSP-based quantum methods both scale as $\tilde{\mathcal{O}}(\epsilon^{-2})$. If the relevant probability distribution of scenarios can be loaded efficiently in superposition, we do not have to load individually sampled scenarios in superposition, and the error ϵ_S in Eq. (42) and Eq. (44) can be made exponentially small, which is not possible with the classical and semi-classical approaches.

VI. COMPARISON OF THE QUANTUM METHODS FOR ESTIMATING VAR

The QAE and QSP VaR quantum estimation methods described in the previous sections are based on the same procedure and it is not obvious whether one outperforms the other in terms of required quantum resources for a target estimation accuracy. The only difference between the two methods is how the probability that a portfolio value under a number of scenarios is smaller than a VaR candidate μ is encoded in an amplitude so that we can perform QAE to read it out. The QAE method performs an analog→digital→analog transformation whereby the portfolio values initially encoded as amplitudes, are digitized approximately into a quantum register by the inner round of QAE before being transformed appropriately back into amplitudes using a binary comparator circuit. On the other hand, the QSP framework allows us to perform the necessary transformation to the amplitudes without the need for an intermediate binary representation which removes that source of error in the process. It does however introduce an error from the approximate implementation of the threshold function which encodes the value to be extracted with QAE. Additionally, looking at the quantum circuits for each method in Fig. 1 and Fig. 3, the QSP circuit is simpler in that a) it only requires one additional qubit other than those required to implement the \mathcal{S} and \mathcal{A} operators which load the scenario superposition and encode the derivative price into an amplitude respectively, b) the \mathcal{A} operator is only controlled on the scenario qubits while it is controlled on one more qubit in the QAE case by virtue of the controlled invocation \mathcal{Q} operator, and c) while the QSP method contains single-qubit phase rotations, the \mathcal{Q} operators in the QAE method require reflections.

The total resources required for both methods can be summarized as

$$C_{\text{VaR}} = C_S C_\mu C_{\text{QAE}} \cdot k, \quad (45)$$

where C_S is the cost of implementing the scenario superposition unitary \mathcal{S} , C_μ denotes the resources required by either method to generate a state where the probability of measuring a qubit in the $|0\rangle$ state gives the probability that the derivative portfolio value falls below a value μ , C_{QAE} is the cost of QAE to read out that probability and k is the number of bisection search iterations required to compute the VaR estimate to a desired accuracy. The cost component C_μ is the main difference between the two methods in terms of the overall algorithmic cost so we focus on estimating the cost of this component in each case. Specifically, we measure the cost by the number of oracle invocations to the pricing operator \mathcal{A} (and its inverse \mathcal{A}^\dagger) which is the most expensive component in derivative pricing [6]. The QAE method requires one invocation of \mathcal{A} to set up the initial state Eq. (10) and m ancilla qubits to discretize the value of the derivative portfolio under the scenarios. Each ancilla qubit $j \in [0, m-1]$ controls 2^j invocations of the \mathcal{Q} operator as shown in Fig. 1, and \mathcal{Q} is comprised of one instance of \mathcal{A} and one of \mathcal{A}^\dagger , giving a total of $N_\mu^{\text{QAE}}(m) = 2^m + 1$ oracle calls. The QSP method uses a d -degree polynomial to approximate the threshold function $\theta_\mu(x)$ of Eq. (22) and its implementation requires $N_\mu^{\text{QSP}}(d) = d - 1$ invocations of \mathcal{A} as shown in Eq. (20) and Fig. 3.

We estimate the values of N_μ^{QAE} and N_μ^{QSP} numerically by sampling simulated scenario prices from a normal distribution with mean 0.5 and standard deviation 0.09 and look for the VaR at the $\alpha = 99\%$ level. The mean and standard deviation of the normal distribution were chosen so that the sampled values lie with high probability in the interval $[0, 1]$ and the $\alpha = 99\%$ confidence level is typical for VaR estimations in practice [32]. We simulate the QAE VaR estimation algorithm as described in Sec. III, where for each sampled scenario price we generate the superposition of binary strings returned by QAE as possible approximations to the value, weighed by the corresponding probabilities of Eq. (14). Starting with $\mu = 0.5$, we calculate the aggregate probability P_0 across all scenarios that the price is smaller than μ . Then, for a parameter ϵ_A we determine bounds $[p_l, p_h] = [P_0 - \epsilon_A, P_0 + \epsilon_A]$ as the confidence interval given by an amplitude estimation round to determine P_0 with target accuracy ϵ_A . We run a bisection search over μ until we find that $1 - \alpha = 0.01 \in [p_l, p_h]$ and we return that value of μ as the VaR estimate.

The QSP method is simulated similarly, the only difference being how we calculate P_0 for a given value of μ . In this case, we pick a value for the polynomial degree d and for each value of μ during bisection search we generate a d -degree polynomial approximation to the threshold function $\theta_\mu(x)$ using the method described in Sec. IV D and apply the resulting function to the sampled scenario prices. Subsequently, we generate the same confidence interval for the amplitude estimation round given a target accuracy ϵ_A as in the QAE method, and impose the same stopping condition. The equivalent classical estimate is computed by sorting the sampled scenario prices and finding the value at the $1 - \alpha$ percentile. Because the simulations of the quantum methods use sampled scenarios, the classical estimate represents the lowest bound either quantum method can achieve.

The results from the simulation for both QAE and QSP methods for different values of oracle call invocations and the final amplitude estimation accuracy ϵ_A are shown in Fig. 6. Each data point is comprised of 100 independent samples, where for each sample we randomly generate $N = 5k$ scenario prices from the normal probability distribution $\mathcal{N}(\mu = 0.5, \sigma^2 = 0.09^2)$, simulate the QAE and QSP algorithms and then compute the error of each sample as the absolute value of the difference between the quantum VaR estimate and the classical VaR estimate from the same generated scenario prices. In the figure we plot the average error across all samples for each data point. One initial observation is that the QSP method performs consistently better for the error range examined, requiring on average 10x fewer oracle calls than QAE for the same target error. In addition, we notice that as the number of oracle calls increase, the probability P' we are estimating each time using amplitude estimation is encoded with increasing accuracy, but clearly at some point this increased encoding accuracy will fail to reduce the VaR estimation error if the value of ϵ_A is not small enough to resolve the difference. This is what we observe at the tail end of the plots in Fig. 6, where the VaR error in the $\epsilon_A = 5 \times 10^{-4}$ plots starts to decrease more slowly than the $\epsilon_A = 10^{-4}$ plots. Until that point, we calculate that for a fixed value of ϵ_A , the VaR error decreases approximately linearly as the number of oracle calls increases for both QAE and QSP cases (the slope of a log-log fit is approximately -0.9 for all four plots), as expected from the error analysis in Sec. V B and Sec. V C respectively where both errors scale as $\mathcal{O}(\epsilon_p^{-1})$ with everything else kept constant.

Our choice of parameters N , μ and σ was made to simplify the numerical simulations, but we observe qualitatively similar results for different choices of distribution parameters μ and σ as well as the number of scenario prices generated for each data point.

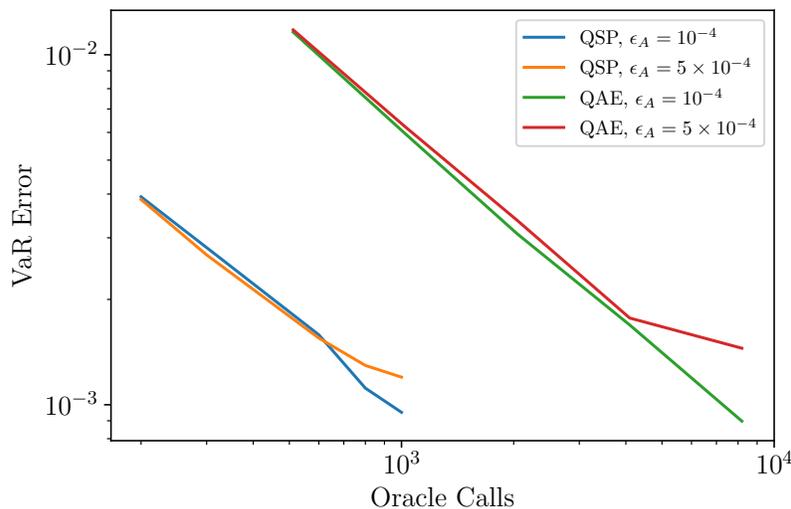


FIG. 6. VaR estimation errors as a function of the number of oracle calls for the QSP and QAE quantum methods. For each data point we generate 100 samples, and for each sample we randomly generate $N = 5k$ scenario prices from the normal probability distribution $\mathcal{N}(\mu = 0.5, \sigma^2 = 0.09^2)$ and compute the VaR estimate by simulating the QAE and QSP algorithms as described in Sec. VI. The error of each sample is calculated as the absolute value of the difference between the quantum VaR estimate and the classical VaR estimate from the same generated scenario prices. We simulate each algorithm for two values of the final amplitude estimation accuracy ϵ_A and plot the average error across all samples for that each data point. On average, we observe that the QSP method requires 10x fewer oracle calls than QAE for the same VaR error. While increasing number of oracle calls N_o leads to the VaR error decreasing as $\mathcal{O}(N_o^{-1})$, notice that this scaling starts to depart from linearity as the amplitude estimation accuracy ϵ_A gets close to the accuracy $1/N_o$ with which the amplitude is encoded. Eventually, the scaling plateaus as the increased encoding accuracy becomes too fine to be resolved by an amplitude estimation process of fixed accuracy.

VII. CONDITIONS FOR QUANTUM ADVANTAGE

On account of the numerical results in the previous section indicating that the QSP-based VaR method outperforms the corresponding QAE variant, in this section we turn our attention on how the VaR estimation with QSP fares against classical methods and what conditions (if any) there are for potential quantum advantage. As discussed in Sec. V, the estimation error of the QSP method is lower-bounded by the classical sampling error of $\mathcal{O}(\epsilon^{-2})$ when the superposition of scenarios is generated with samples from a probability distribution. Assuming no particular structure in the representation of the sampled scenarios, the superposition of N scenarios can be loaded to accuracy ϵ using the QRAM-based state preparation routine from Ref. [31] with T-depth $\mathcal{O}(\log(N) + \log(1/\epsilon))$, at a cost of $\mathcal{O}(N)$ T-count. On the other hand, if the scenarios can be prepared by discretizing a multivariate probability distribution of the relevant risk factors and loaded efficiently in superposition, the lower bound on the VaR estimation error can be made exponentially small by increasing the number of qubits representing the distribution. In the case that the probability distribution can be modeled explicitly with an analytic density function, the scenario superposition could be prepared using the re-parameterization method from Ref. [6]. Alternatively, if the scenario probabilities have to be inferred from historical data, quantum representations of financially relevant copulas [33, 34] could potentially be employed if they can scale to scenarios of high dimensionality. In either case, the resources required to generate the scenario superposition will be highly dependent on the nature of the target probability distribution if explicit loading is possible, and the number of risk factors that make up each scenario as well as the required granularity for each risk factor (as shown in Eq. (7)). As such, in this section we do not examine specific choices of scenario preparation methods, and instead consider the cost of loading the superposition to be a free variable which needs to be taken into account when estimating the total resources for the algorithm.

As a benchmark to examine the conditions for practical quantum advantage, we consider the autocallable derivative product studied in Ref. [6]. We calculate the resources required by the QSP-based VaR algorithm to estimate the VaR of the product to the same accuracy as a classical Monte Carlo method, assuming that the product is evaluated under scenarios such that the resulting prices are distributed normally. We use the latest available resource estimates from Ref. [19], whereby an autocallable derivative product can be priced to within $\epsilon_p = 2 \times 10^{-3}$ in one second classically and the \mathcal{A} operator required to price the contract on a quantum computer can be constructed with a T-depth of

$T_A = 3900$.

The pricing of this product under N scenarios is simulated classically by randomly generating N values from a normal distribution $\mathcal{N}(\mu, \sigma^2)$ representing N scenario prices, and adding a noise term sampled from $\mathcal{N}(0, \epsilon_p^2)$. Then we compute V_α from Eq. (1) by sorting the resulting simulated prices from smallest to largest, and picking the value at the $1 - \alpha$ percentile. The error of the VaR estimate from the classical simulation can be calculated as $|V_\alpha - \Phi^{-1}(1 - \alpha)|$, where $\Phi(x)$ is the cumulative distribution function of the distribution $\mathcal{N}(\mu, \sigma^2)$. We repeat this process 200 times, and define the error of the classical algorithm ϵ_C as the standard deviation of the resulting distribution of errors. Since we assume the contract can be priced classically in one second, for N scenarios, we consider the runtime of the end-to-end classical VaR estimation to be N seconds.

To evaluate the performance of the quantum QSP VaR algorithm, we similarly generate N values from the normal distribution $\mathcal{N}(\mu, \sigma^2)$ representing N scenario prices and simulate the QSP algorithm as described in Algorithm 1. The error of the quantum estimate is given by $|\mu_\alpha^2 - \Phi^{-1}(1 - \alpha)|$, where μ_α is the estimate of $\sqrt{V_\alpha}$ returned by the QSP algorithm. Repeating the process 200 times, we define the error of the quantum algorithm ϵ_Q as the value at the 68th percentile of the resulting distribution of errors. This way we can say that with probability 68% the estimation error of the algorithm does not exceed ϵ_Q , which corresponds to the same confidence in the estimate of the classical algorithm, defined as the error at one standard deviation.

We then proceed to estimate the T-depth of the end-to-end process, as a proxy for the total runtime. The T-depth of each iteration, which is the T-depth of the circuit of Fig. 3 is given by

$$T_i = T_S + dT_A + dT_R, \quad (46)$$

where T_S is the T-depth of the scenario preparation unitary \mathcal{S} , $T_A = 3900$ is the T-depth of the \mathcal{A} operator [19], d is the polynomial degree used for the approximation to the threshold function and T_R is the T-depth of implementing each controlled R_z rotation. Using the method in [35], the R_z rotation can be performed to precision ϵ_R with a T-depth of approximately $3 \log_2(1/\epsilon_R)$ and the decomposition in [36] allows us to perform the controlled rotation with an R_z depth of one, using one ancilla qubit. We measure the cost of each round of QAE for target accuracy $\epsilon_A > 0$ and confidence level $1 - \alpha_k$, $\alpha_k \in (0, 1)$ by the bound derived in [22]

$$N_{\text{oracle}}^{\text{wc}} \leq \frac{1.4}{\epsilon_A} \log \left(\frac{2}{\alpha_k} \log_2 \left(\frac{\pi}{4\epsilon_A} \right) \right), \quad (47)$$

where $N_{\text{oracle}}^{\text{wc}}$ denotes the worst-case number of QAE oracle calls. In our case, the QAE oracle call consists of one invocation of the entire circuit in Fig. 3 and one to its inverse (where we ignore the cost of reflections in the QAE oracle). Thus, the end-to-end T-depth of the QSP VaR algorithm which goes through k rounds of QAE is given by

$$T_{\text{QSP}} = \frac{2.8k}{\epsilon_A} \log \left(\frac{2}{\alpha_k} \log_2 \left(\frac{\pi}{4\epsilon_A} \right) \right) (T_S + dT_A + 3d \log_2(1/\epsilon_R)). \quad (48)$$

Additionally, we choose ϵ_R such that the total error from all the R_z rotations is the same order of magnitude as the error from the polynomial approximation to the threshold function, which is $\gtrsim 10^{-4}$ as shown in Fig. 5. In our numerical simulations the polynomial degrees we use satisfy $d \leq 1000$, so we pick $\epsilon_R = 10^{-7}$.

The remaining free parameters in the total cost of the QSP algorithm are the scenario loading cost T_S , the polynomial degree d used for QSP and the target accuracy ϵ_A with which we estimate the encoded probability in each round of QAE. Additionally, for a fixed number of scenarios N , the VaR estimation error is only dependent on the choices of d and ϵ_A . In order to benchmark the performance of the QSP algorithm, for a choice of N , we numerically look for the optimal values of parameters (d, ϵ_A) such that the estimation error of the quantum algorithm ϵ_Q is equal to that of the classical algorithm ϵ_C and calculate the overall T-depth given by Eq. (48) with T_S left as the only remaining free cost parameter. Once we estimate the optimal values of (d, ϵ_A) , we can additionally define the minimum rate at which that a quantum processor would need to execute T-gates so that the physical runtime of the QSP method would equal the physical runtime of a classical processor. That *logical* QPU clock rate is defined as $L \equiv T_{\text{QSP}}/N$, where, as described above, we assume it takes N seconds to classically price N scenarios.

We use $\mu = 0.5$ and $\sigma = 0.09$ as the parameters of the normal distribution from which we draw scenario samples, and simulate the classical VaR estimation method for $\alpha = 99\%$. The QSP algorithm is simulated for the same value of σ , but in this case we average the results over $\mu \in [0.45, 0.48, 0.5, 0.52, 0.55]$. The reason for this is that the QSP method can do better (or worse) depending on where the true VaR value lies, because of the bisection search performed. Specifically, the estimation error of the quantum method will depend on the distance between the true VaR value and the values reachable by bisection search in the interval $[0, 1]$. We therefore use this averaging over

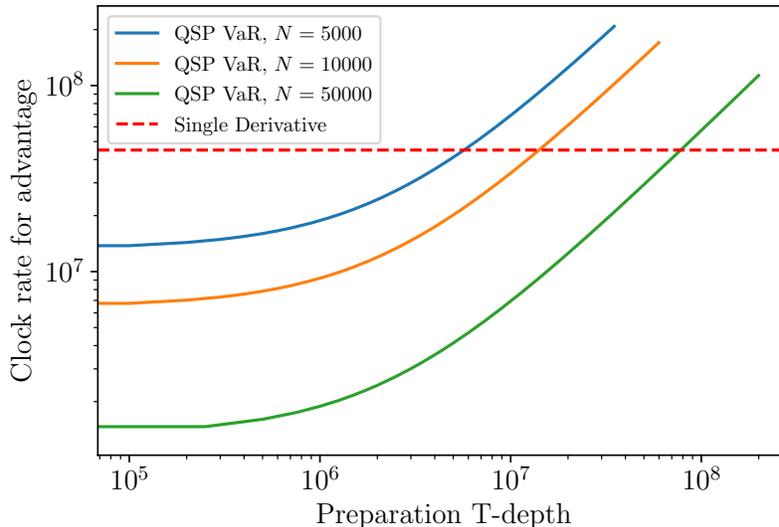


FIG. 7. Minimum QPU logical clock rate required for the QSP VaR method to match classical methods in total execution time as a function of the scenario loading T-depth. For reference, the horizontal dashed line indicates the minimum logical clock rate required for quantum advantage in the pricing of a single derivative as estimated in Ref. [19]. Both classical and quantum algorithms are simulated by sampling scenarios from a normal distribution as described in Sec. VII, and for each choice of number of scenarios N , we search for the parameters (d, ϵ_A) (controlling the polynomial approximation to the threshold function and the accuracy of QAE respectively) such that the estimation error from the QSP algorithm matches that of the classical algorithm. For $N = 50k$, we see that there is a possibility for reduction of the logical clock rate required for quantum advantage for up to a factor $\sim 30x$ compared to that required in the pricing of a single derivative. For all choices of N shown here, we numerically estimate that the QSP method requires at least $d = 600$ and $\epsilon_A = 1.2 \times 10^{-3}$ in order to match the corresponding estimation error of the classical method.

values of μ to benchmark the average performance of the algorithm. On the other hand, the estimation error has no dependence on the true value of VaR classically, so we fix the value of μ in that case. We chose $\sigma = 0.09$ for the simulated scenario distribution such that the sampled scenarios are in the interval $[0, 1]$ with high probability. We have also simulated the algorithms for $\sigma \in [0.05, 0.07]$ and our results remain qualitatively unchanged.

In Fig. 7 we show the logical clock rate required for quantum advantage as estimated from the numerical simulations, as function of the scenario preparation T-depth T_S , for different numbers of scenarios N . For reference, we also show the logical clock rate (45Mhz) required for quantum advantage in pricing a single derivative calculated in Ref. [19]. Because the runtime of the classical and semi-classical methods scales linearly with N , as N increases, the QSP method starts to provide room for quantum advantage as long as the scenario preparation can be prepared with a smaller T-depth than a certain threshold, which can be derived from Eq. (48). For small preparation T-depths, the value of T_S is dominated by the resources required for QSP, reducing the logical clock rate for advantage until the two terms become comparable. For $N = 50k$, we notice that there is an advantage over single derivative pricing as long as the scenario superposition can be prepared with a T-depth $T_S \lesssim 8 \times 10^7$. In the case that the superposition can be prepared with a T-depth $T_S \lesssim 3 \times 10^5$, the logical clock rate required for advantage is reduced by a factor of $\sim 30x$.

We note here that these estimates for quantum advantage are based on the assumption that the superposition over scenarios is created from sampled scenarios. In the case where an explicit probability distribution of scenarios can be loaded efficiently as a discretized probability density function, the error from the scenario sampling would decrease exponentially and allow the possibility of further quantum advantage.

VIII. DISCUSSION

While the efficacy of quantum computers in derivative pricing has been studied in the context of estimating the value of individual derivatives, in various relevant contexts, the quantity of interest is not the individual derivative price itself, but rather metrics which characterize the risk profile of financial derivatives. Because calculating these metrics often requires pricing derivatives repeatedly, the power of quantum computers could be harnessed more effectively by considering the aggregate computation instead of the application of quantum computing to the individual

components. As such, devising quantum algorithms which use the oracles constructed for the pricing of derivatives as subcomponents, opens up a new avenue for quantum advantage in the context of derivative pricing [7]. In this work, we consider the problem of estimating the Value at Risk and the Conditional Value at Risk of financial derivatives. We first extend the algorithm introduced in Ref. [3, 10] to estimate the VaR of financial derivatives and then introduce a quantum algorithm based on the Quantum Signal Processing framework to estimate these metrics. Even though we show that asymptotically these two algorithms have similar performance, we show that the use of QSP in practice allows for a more efficient algorithm, a conclusion which has also been observed in different contexts [16–18]. This result also highlights the practical advantage of QSP-based methods which not only allow the application of generic transformations to appropriately encoded values of interest, but that they do so with minimal overhead. While our formulation of derivative pricing in the QSP framework was used in this case to calculate the VaR and CVaR risk metrics, we expect that it will open up new research avenues in the larger context of derivative pricing using quantum computers.

Additionally, we study the possibility of quantum advantage by using this algorithm over equivalent classical methods. The performance of the algorithm relies on the ability to prepare a superposition of scenarios efficiently and different approaches to that end have been proposed [6, 33, 34]. During the writeup of this manuscript, a state preparation method was introduced to load a uniform superposition of subsets of computational basis states scaling as $\mathcal{O}(\log_2 M)$ for M states in superposition, with low overhead [37]. The superposition over scenarios encoded using Eq. (7) is precisely a question of generating a uniform superposition over subsets of computational basis states, and therefore determining the applicability of this method to the VaR algorithms we describe here is a very promising research direction. In this work, we have left the scenario preparation as an open question and instead, through numerical simulations, we estimate upper bounds on the resources required to prepare the necessary superposition in order for quantum advantage to exist. We find that in certain settings, the quantum algorithm can provide a reduction in the logical clock rate required for quantum advantage estimated in Ref. [19] by $\sim 30x$. While this estimate is based on pricing the derivative under scenarios sampled from a relevant probability distribution, quantum computing allows the possibility of using the full (discretized) multivariate probability distribution in the calculation, an option not available to classical methods due to the curse of dimensionality. An extension of this work would be to study which (relevant) probability distributions can be loaded efficiently on a quantum computer and the impact on the estimate of possible quantum advantage calculated in this work.

In our comparison between the performance of quantum and classical methods, we have assumed that classical methods for the VaR estimation of financial derivatives require pricing derivative contracts across a fixed number of scenarios N , such that the total complexity scales as $\mathcal{O}(N/\epsilon^2)$. In Ref. [38], the authors argue that for sufficiently smooth probability density functions, an adaptive sampling technique combined with Multi-level Monte Carlo can reduce the complexity of general VaR and CVaR estimation to $\tilde{\mathcal{O}}(\epsilon^{-2})$. One core element in that approach is that the probability distribution behind the scenarios used in the VaR estimation can be easily sampled, such that the number of samples generated can be adjusted to fit the target accuracy of the calculation. However, in practice, it is extremely difficult to generate such probability distribution across (typically) thousands of market factors, and most commonly a fixed number of scenarios is created based on historical data that capture market features deemed important from a financial modeler’s perspective. That said, Multi-level Monte Carlo can also be applied in a quantum setting [39], and adaptive samples can similarly be incorporated in the quantum process, by adjusting the accuracy with which the derivative price is encoded as a quantum amplitude. We leave the detailed analysis of this approach to a future study.

ACKNOWLEDGMENTS

We thank Bryce Fuller, Patrick Rall and Farrokh Labib for discussions regarding Quantum Signal Processing and feedback on this manuscript, Kunal Kishore for his technical and business insights regarding the VaR estimation of financial derivatives, and Noelle Ibrahim for discussions on classical adaptive methods for estimating VaR.

-
- [1] A. Montanaro, “Quantum speedup of Monte Carlo methods,” *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* **471** (2015).
 - [2] P. Rebentrost, B. Gupt, and T. R. Bromley, “Quantum computational finance: Monte Carlo pricing of financial derivatives,” *Phys. Rev. A* **98**, 022321 (2018).
 - [3] S. Woerner and D. J. Egger, “Quantum risk analysis,” *npj Quantum Information* **5** (2019).
 - [4] N. Stamatopoulos, D. J. Egger, Y. Sun, C. Zoufal, R. Iten, N. Shen, and S. Woerner, “Option Pricing using Quantum Computers,” *Quantum* **4**, 291 (2020).

- [5] S. Herbert, “Quantum Monte Carlo Integration: The Full Advantage in Minimal Circuit Depth,” *Quantum* **6**, 823 (2022).
- [6] S. Chakrabarti, R. Krishnakumar, G. Mazzola, N. Stamatopoulos, S. Woerner, and W. J. Zeng, “A Threshold for Quantum Advantage in Derivative Pricing,” *Quantum* **5**, 463 (2021).
- [7] N. Stamatopoulos, G. Mazzola, S. Woerner, and W. J. Zeng, “Towards Quantum Advantage in Financial Market Risk using Quantum Gradient Algorithms,” *Quantum* **6**, 770 (2022).
- [8] H. Markowitz, “Portfolio Selection,” *The Journal of Finance* **7**, 77 (1952).
- [9] A. D. Roy, “Safety First and the Holding of Assets,” *Econometrica* **20**, 431 (1952).
- [10] D. J. Egger, R. G. Gutierrez, J. C. Mestre, and S. Woerner, “Credit risk analysis using quantum computers,” *IEEE Transactions on Computers* (2020).
- [11] G. H. Low and I. L. Chuang, “Optimal Hamiltonian Simulation by Quantum Signal Processing,” *Phys. Rev. Lett.* **118**, 010501 (2017).
- [12] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, “Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics,” in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (2019) pp. 193–204.
- [13] G. H. Low and I. L. Chuang, “Hamiltonian simulation by qubitization,” *Quantum* **3**, 163 (2019).
- [14] Y. Kikuchi, C. Mc Keever, L. Coopmans, M. Lubasch, and M. Benedetti, “Realization of quantum signal processing on a noisy quantum computer,” *npj Quantum Information* **9** (2023).
- [15] J. M. Martyn, Z. M. Rossi, A. K. Tan, and I. L. Chuang, “Grand Unification of Quantum Algorithms,” *PRX Quantum* **2** (2021).
- [16] J. M. Martyn, Y. Liu, Z. E. Chin, and I. L. Chuang, “Efficient Fully-Coherent Hamiltonian Simulation,” arXiv preprint arXiv:2110.11327 (2021), arXiv:2110.11327 [quant-ph].
- [17] L. Lin and Y. Tong, “Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems,” *Quantum* **4**, 361 (2020).
- [18] P. Rall and B. Fuller, “Amplitude Estimation from Quantum Signal Processing,” *Quantum* **7**, 937 (2023).
- [19] N. Stamatopoulos and W. J. Zeng, “Derivative Pricing using Quantum Signal Processing,” (2023), arXiv:2307.14310 [quant-ph].
- [20] A. Gilyén, S. Arunachalam, and N. Wiebe, “Optimizing quantum optimization algorithms via faster quantum gradient computation,” *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1425–1444 (2019).
- [21] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, “Quantum Amplitude Amplification and Estimation,” *Contemporary Mathematics* **305** (2002).
- [22] D. Grinko, J. Gacon, C. Zoufal, and S. Woerner, “Iterative quantum amplitude estimation,” *npj Quantum Information* **7** (2021).
- [23] Y. Suzuki, S. Uno, R. Raymond, T. Tanaka, T. Onodera, and N. Yamamoto, “Amplitude estimation without phase estimation,” *Quantum Information Processing* **19**, 75 (2020).
- [24] T. Giurgica-Tiron, I. Kerenidis, F. Labib, A. Prakash, and W. Zeng, “Low depth algorithms for quantum amplitude estimation,” *Quantum* **6**, 745 (2022).
- [25] P. Rall, “Faster Coherent Quantum Algorithms for Phase, Energy, and Amplitude Estimation,” *Quantum* **5**, 566 (2021).
- [26] J. Haah, “Product Decomposition of Periodic Functions in Quantum Signal Processing,” *Quantum* **3**, 190 (2019).
- [27] R. Chao, D. Ding, A. Gilyen, C. Huang, and M. Szegedy, “Finding Angles for Quantum Signal Processing with Machine Precision,” arXiv preprint arXiv:2003.02831 (2020), arXiv:2003.02831 [quant-ph].
- [28] Y. Dong, X. Meng, K. B. Whaley, and L. Lin, “Efficient phase-factor evaluation in quantum signal processing,” *Physical Review A* **103**, 042419 (2021).
- [29] Y. Dong, L. Lin, and Y. Tong, “Ground-State Preparation and Energy Estimation on Early Fault-Tolerant Quantum Computers via Quantum Eigenvalue Transformation of Unitary Matrices,” *PRX Quantum* **3** (2022).
- [30] G. H. Low and I. L. Chuang, “Hamiltonian simulation by uniform spectral amplification,” arXiv preprint arXiv:1707.05391 (2017).
- [31] B. D. Clader, A. M. Dalzell, N. Stamatopoulos, G. Salton, M. Berta, and W. J. Zeng, “Quantum Resources Required to Block-Encode a Matrix of Classical Data,” *IEEE Transactions on Quantum Engineering* **3**, 1–23 (2022).
- [32] J. O’Brien and P. J. Szerszeń, “An evaluation of bank measures for market risk before, during and after the financial crisis,” *Journal of Banking & Finance* **80**, 215 (2017).
- [33] J. Milek, “Quantum Implementation of Risk Analysis-relevant Copulas,” arXiv preprint arXiv:2002.07389 (2020).
- [34] D. Zhu, W. Shen, A. Giani, S. R. Majumder, B. Neclaes, and S. Johri, “Copula-based Risk Aggregation with Trapped Ion Quantum Computers,” arXiv preprint arXiv:2206.11937 (2022).
- [35] N. J. Ross and P. Selinger, “Optimal Ancilla-Free Clifford+T Approximation of z-Rotations,” *Quantum Info. Comput.* **16**, 901 (2016).
- [36] T. Kim and B. Choi, “Efficient decomposition methods for controlled-R_n using a single ancillary qubit,” *Scientific Reports* **8** (2018).
- [37] A. Shukla and P. Vedula, “An efficient quantum algorithm for preparation of uniform quantum superposition states,” *Quantum Information Processing* **23** (2024).
- [38] M. B. Giles and A.-L. Haji-Ali, “Multilevel Nested Simulation for Efficient Risk Estimation,” *SIAM/ASA Journal on Uncertainty Quantification* **7**, 497 (2019).
- [39] D. An, N. Linden, J.-P. Liu, A. Montanaro, C. Shao, and J. Wang, “Quantum-accelerated multilevel Monte Carlo methods for stochastic differential equations in mathematical finance,” *Quantum* **5**, 481 (2021).

Appendix A: Derivative Pricing using Amplitude Estimation

The price of a derivative contract is calculated as the expectation value of the derivative's discounted payoff, evaluated over a suitable stochastic process that is assumed to govern the dynamics of the underlying market parameters. Given stochastic paths $\omega \in \Omega$, each of which occurring with probability $p(\omega)$, the expectation value of the discounted payoff $f(\omega)$ can be written as

$$\mathbb{E}[f] = \sum_{\omega \in \Omega} p(\omega) f(\omega). \quad (\text{A1})$$

Quantum Amplitude Estimation (QAE) [1, 21] can be used to estimate such expectation values if the quantity of interest can be efficiently encoded as the probability of a specific measurement outcome. QAE takes as input a unitary operator \mathcal{A} which produces the state

$$\mathcal{A} : |0\rangle_{n+1} \rightarrow \left(\sqrt{V} |\psi_0\rangle_n |0\rangle + \sqrt{1-V} |\psi_1\rangle_n |1\rangle \right), \quad (\text{A2})$$

where $|\psi_0\rangle$ and $|\psi_1\rangle$ are arbitrary, normalized quantum states and estimates the derivative price V to additive error ϵ using $\mathcal{O}(1/\epsilon)$ invocations of the operator $\mathcal{Q} = \mathcal{A}S_0\mathcal{A}^\dagger S_{\psi_1}$, where $S_0 = \mathbb{I} - 2|0\rangle_{n+1}\langle 0|_{n+1}$ and $S_{\psi_1} = \mathbb{I} - 2|1\rangle_{\psi_1}\langle 1|_{\psi_1}$.

A method to construct the \mathcal{A} operator of Eq. (A2) for typical derivative contracts of practical interest and the corresponding required quantum resources is shown in [6]. For a contract with d stochastic market parameters $X = [x_1, x_2, \dots, x_d]$, an operator \mathcal{P} constructs a probability-weighted superposition of all possible combinations of stochastic realizations of the market parameters,

$$\mathcal{P} : |\vec{0}\rangle^{\otimes d} \rightarrow \sum_{\omega} \sqrt{p(\omega)} |\omega\rangle, \quad (\text{A3})$$

with an appropriate discretization for each parameter chosen such that the overall error from such discretization satisfies a desired accuracy. Typically, each market parameter is modeled stochastically at pre-defined points in time that are determined by the definition of the derivative contract. As such, each $|\omega\rangle$ contains all the information about the values of the market parameters X for each *path* of the stochastic process. Then, an operator \mathcal{F} computes the discounted payoff f of the derivative on each path $|\omega\rangle$, and encodes that value in the amplitude of an ancilla qubit

$$\mathcal{F} : |\omega\rangle |\vec{0}\rangle |0\rangle \rightarrow |\omega\rangle |f(\omega)\rangle \left(\sqrt{f(\omega)} |0\rangle + \sqrt{1-f(\omega)} |1\rangle \right). \quad (\text{A4})$$

Taking $\mathcal{A} = \mathcal{F}\mathcal{P}$

$$\mathcal{F}\mathcal{P} : |\vec{0}\rangle^{\otimes d} |0\rangle \rightarrow \left(\sum_{\omega} \sqrt{p(\omega)} \sqrt{f(\omega)} |\omega\rangle |f(\omega)\rangle |0\rangle + \sum_{\omega} \sqrt{p(\omega)} \sqrt{1-f(\omega)} |\omega\rangle |f(\omega)\rangle |1\rangle \right), \quad (\text{A5})$$

gives us the desired operator of Eq. (A2), noticing that the probability of measuring $|0\rangle$ in the last qubit is the expectation value of the discounted payoff, as defined in Eq. (A1).

This approach can be generalized to compute the value V of a derivative portfolio consisting of m derivatives. The \mathcal{P} operator of Eq. (A3) in this case will construct the stochastic process of the d market parameters of the entire portfolio, and the \mathcal{F} operator of Eq. (A4) will compute the sum of the payoffs of each derivative

$$\mathcal{F} : |\omega\rangle |\vec{0}\rangle |0\rangle \rightarrow |\omega\rangle \left| \sum_{i=1}^m f_i(\omega) \right\rangle \left(\sqrt{\sum_{i=1}^m f_i(\omega)} |0\rangle + \sqrt{1 - \sum_{i=1}^m f_i(\omega)} |1\rangle \right), \quad (\text{A6})$$

where f_i is the payoff of derivative $i \in [1, m]$. Similarly to the single-derivative case, QAE can then be used to estimate the expectation value V of the portfolio.

$$V = \sum_{\omega \in \Omega} p(\omega) \sum_{i=1}^m f_i(\omega). \quad (\text{A7})$$