Derivative Pricing using Quantum Signal Processing

Nikitas Stamatopoulos and William J. Zeng

Goldman Sachs, New York, NY

Pricing financial derivatives on quantum computers typically includes quantum arithmetic components which contribute heavily to the quantum resources required by the corresponding circuits. In this manuscript, we introduce a method based on Quantum Signal Processing (QSP) to encode financial derivative payoffs directly into quantum amplitudes, alleviating the quantum circuits from the burden of costly quantum arithmetic. Compared to current state-of-the-art approaches in the literature, we find that for derivative contracts of practical interest, the application of QSP significantly reduces the required resources across all metrics considered, most notably the total number of T-gates by $\sim$ 16x and the number of logical qubits by $\sim$ 4x. Additionally, we estimate that the logical clock rate needed for quantum advantage is also reduced by a factor of $\sim$ 5x. Overall, we find that quantum advantage will require 4.7k logical qubits, and quantum devices that can execute 10^9 T-gates at a rate of 45MHz. While in this work we focus specifically on the payoff component of the derivative pricing process where the method we present is most readily applicable, similar techniques can be employed to further reduce the resources in other applications, such as state preparation.

1 Introduction

Financial derivatives can be priced on quantum computers using Quantum Amplitude Estimation (QAE) [1] with a quadratic speedup compared to classical Monte Carlo methods [2–7]. However, the resources required by the resulting quantum circuits have been estimated to be considerably larger than what is expected to be possible with near term devices [8]. Possible reduction of these resources has been proposed by replacing quantum arithmetic with lookup tables [9], applying generative methods to prepare relevant probability distributions [10], extracting multiple other useful quantities in derivative pricing [11], and improving amplitude estimation [12, 13].

Quantum Signal Processing[1] (QSP) is a technique to perform nearly arbitrary polynomial transformations to quantum (sub)systems with low resource overhead [14]. Various quantum algorithms have been unified under the QSP framework [15] and in multiple instances the QSP formulation has led to a significant reduction in resource overheads [16–20]. In particular, in Ref. [20] the authors show how to apply QSP for quantum state preparation in cases where the target amplitudes are known functions of a variable whose discretized values have been encoded as computational basis states. Additionally, the authors demonstrated that this QSP approach significantly outperforms the use of quantum (coherent) arithmetic for the same task in terms of the total quantum resources required, with most pronounced impact on the number of qubits.

Because the quantum oracles involved in derivative pricing inevitably include components heavily relying on quantum arithmetic [8], in this manuscript we show how QSP and the ideas from Ref. [20] can be used to reduce the quantum resources needed for derivative pricing for certain very common types of derivatives. We identify the specific circuit components where QSP can replace quantum arithmetic routines, and provide implementations for all the unitaries involved across the entire applicable domain. Moreover, we introduce a new unitary to drive the QSP process, leading

[1] This technique is also referred to as Quantum Singular Value Transformation (QSVT) or Quantum Eigenvalue Transform (QET) depending on the shape/dimensionality of the subsystem transformed. In this manuscript we use QSP as an umbrella term encompassing all transformations of this nature which rely on the same underlying process.

to further reduction in resources compared to that considered in Ref. [20]. We examine the impact of this method on concrete derivative applications of practical interest and demonstrate the precise impact on quantum resources by explicitly constructing the corresponding circuits.

The rest of the paper in structured as follows: In Sec. 2 we describe the core components used for derivative pricing on a quantum computer and highlight the bottlenecks we target in this work. In Sec. 3 we outline the QSP framework and present the core ideas used to apply the framework to derivative pricing. Then, in Sec. 4 we describe the implementation details of the proposed QSP approach and in Sec. 5 we demonstrate how to apply the ideas developed to derivative pricing instances of practical interest. In Sec. 6 we calculate the quantum resources needed for quantum advantage in derivative pricing using the method we develop, and contrast them with those of previous approaches in the literature. Finally, in Sec. 7, we discuss the implications of this work, highlight its limitations, and outline potential future research directions on the topic.

2 Derivative Pricing with Quantum Computers

Financial derivatives are contracts whose value depends on the future performance of one or more underlying assets. Derivative contracts are typically defined by their *payoff*, which determines the value of the contract for a given future *path* of the underlying assets. Classically, these contracts are often priced by simulating a large number of possible such paths according to an appropriate stochastic model, computing the payoff for each simulated path, and calculating the expectation value across all future payoffs. The *fair price* of the derivative contract is then given by that future expectation value, appropriately discounted to take into account the difference between the time the contract is priced and its future value. For an overview of derivative types, features and classical pricing methodologies, we refer the reader to Ref. [21].

In a quantum setting, the quantum circuits which drive the pricing process consist of two main components [8]: 1) loading the probability distribution over paths of the (discretized) stochastic variables involved in the calculation and 2) encoding the payoff of each path into the amplitude of an ancilla qubit. The first component can be represented by an operator $\mathcal{P}$ which, given stochastic paths $\omega \in \Omega$ where each path occurs with probability $p(\omega)$, it prepares a probability-weighted superposition of all paths

$$\mathcal{P} : |\vec{0}\rangle \rightarrow \sum_{\omega} \sqrt{p(\omega)}\, |\omega\rangle \,. \tag{1}$$

The second component is an operator $\mathcal{F}$ which computes the payoff $f(\omega)$ of the derivative on each path $|\omega\rangle$, and encodes that value in the amplitude of an ancilla qubit

$$\mathcal{F} : |\omega\rangle\, |0\rangle \rightarrow |\omega\rangle\, |f(\omega)\rangle \left(\sqrt{f(\omega)}\, |0\rangle + \sqrt{1 - f(\omega)}\, |1\rangle \right). \tag{2}$$

The operator $\mathcal{A} = \mathcal{F}\mathcal{P}$

$$\mathcal{A} : |\vec{0}\rangle\, |0\rangle \rightarrow \left(\sum_{\omega} \sqrt{p(\omega)} \sqrt{f(\omega)}\, |\omega\rangle\, |f(\omega)\rangle\, |0\rangle + \sum_{\omega} \sqrt{p(\omega)} \sqrt{1 - f(\omega)}\, |\omega\rangle\, |f(\omega)\rangle\, |1\rangle \right), \tag{3}$$

creates a state such that the probability of measuring $|0\rangle$ in the last qubit is the price of the derivative, represented as the expected value of the payoff, $\mathbb{E}[f] = \sum_{\omega \in \Omega} p(\omega) f(\omega)$.

The implementation of the $\mathcal{F}$ operator of Eq. (2) requires binary quantum arithmetic to compute a digital representation of the payoff $|f(\omega)\rangle$ from a path $|\omega\rangle$, before that value is encoded into the amplitude of the last qubit. It is this component we will aim to implement using QSP, such that the value $\sqrt{f(\omega)}$ is directly encoded in the amplitude of an ancilla qubit without requiring the computation of $|f(\omega)\rangle$, thus eliminating the costly quantum arithmetic involved.

3 Quantum Signal Processing

Given a quantum register $|x\rangle_n$ which stores an n-bit representation of a scalar $x \in [0, 1]$, a real function $g : [0, 1] \rightarrow [0, 1]$ and a unitary which creates the state

$$U\left|x\right\rangle_n\left|0\right\rangle_{n+1} = \left|x\right\rangle_n\left(g(x)\left|\psi_0\right\rangle_n\left|0\right\rangle + \sqrt{1-g(x)^2}\left|\psi_1\right\rangle_n\left|1\right\rangle\right), \tag{4}$$

where $\left|\psi_0\right\rangle_n$ and $\left|\psi_1\right\rangle_n$ are normalized quantum states, Quantum Signal Processing (QSP) can be used to apply polynomial transformations to the amplitude $g(x)$.

In Ref. [14], the authors observe that given projectors $\tilde{\Pi} \equiv \mathbb{I}^{\otimes 2n} \otimes \left|0\right\rangle\left\langle 0\right|$ and $\Pi \equiv \mathbb{I}^{\otimes n} \otimes \left(\left|0\right\rangle\left\langle 0\right|\right)^{\otimes n+1}$, U is a block-encoding of the rank-1 matrix $A = \tilde{\Pi}U\Pi$

$$U = \begin{array}{c} \Pi \\ \tilde{\Pi} \end{array}\begin{bmatrix} A & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{5}$$

with a single non-trivial singular value $g(x)$. They then show that consecutive invocations of U and $U^\dagger$, interleaved with projector-controlled rotation operators $\Pi_\phi = e^{i\phi(2\Pi-\mathbb{I})}$ and $\tilde{\Pi}_\phi = e^{i\phi(2\tilde{\Pi}-\mathbb{I})}$, can be used to apply polynomial transformations to this singular value. Specifically, given phase factors $\vec{\phi} = (\phi_1, \phi_2, \cdots, \phi_d)$, define

$$\mathcal{U}^{\vec{\phi}} = \begin{cases} \tilde{\Pi}_{\phi_1} U \displaystyle\prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U^\dagger \tilde{\Pi}_{\phi_{2k+1}} U, & \text{for odd } d, \\ \displaystyle\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U^\dagger \tilde{\Pi}_{\phi_{2k}} U, & \text{for even } d \end{cases} \tag{6}$$

and

$$\mathcal{U}_C^{\vec{\phi}} = \left(\mathcal{U}^{\vec{\phi}} \otimes \left|0\right\rangle\left\langle 0\right| + \mathcal{U}^{-\vec{\phi}} \otimes \left|1\right\rangle\left\langle 1\right|\right). \tag{7}$$

Then, for any polynomial P satisfying $\deg(P) \leq d$, $|P(a)| \leq 1, \forall a \in [-1,1]$, and P either even or odd, the authors of Ref. [14] show that there exist phase factors $\vec{\phi}$ [22–24] such that the unitary

$$U^{\vec{\phi}} = (\mathbb{I}^{\otimes 2n+1} \otimes H)\mathcal{U}_C^{\vec{\phi}}(\mathbb{I}^{\otimes 2n+1} \otimes H), \tag{8}$$

applies the polynomial P to the singular values of matrix A

$$U^{\vec{\phi}} = \begin{array}{c} \left(\Pi \otimes \left|0\right\rangle\left\langle 0\right|\right) \\ \left(\Pi' \otimes \left|0\right\rangle\left\langle 0\right|\right) \end{array}\begin{bmatrix} P(A) & \cdot \\ \cdot & \cdot \end{bmatrix} \tag{9}$$

where $\Pi' = \tilde{\Pi}$ for odd d and $\Pi' = \Pi$ for even d. Equivalently, $U^{\vec{\phi}}$ creates the state

$$U^{\vec{\phi}}\left|x\right\rangle_n\left|0\right\rangle_{n+1}\left|0\right\rangle = \left|x_n\right\rangle \begin{cases} \left(P(g(x))\left|\psi_0'\right\rangle_n\left|0\right\rangle_2 + \sqrt{1-P(g(x))^2}\left|\psi_1'\right\rangle_n\left|0_\perp\right\rangle_2\right), & \text{for odd } d, \\ \left(P(g(x))\left|0\right\rangle_{n+2} + \sqrt{1-P(g(x))^2}\left|0_\perp\right\rangle_{n+2}\right), & \text{for even } d. \end{cases} \tag{10}$$

A circuit description of the operator $U^{\vec{\phi}}$ is shown in Fig. 1.

In Ref. [20], the authors consider the scenario where a target function $f(x)$ is computed by first constructing a unitary of the form of Eq. (4) and then applying a polynomial transformation using QSP approximating the function composition $\left(f \circ g^{-1}\right)(x)$. Specifically, the unitary in Eq. (4) is taken to be

$$U_{\sin}\left|x\right\rangle_n\left|0\right\rangle = \left|x\right\rangle_n\left(\sin(x)\left|0\right\rangle + \cos(x)\left|1\right\rangle\right), \tag{11}$$

and QSP transformations based on this unitary are used to prepare states whose amplitudes are functions of x, without requiring quantum arithmetic. With this unitary the polynomial transformation applied with QSP needs to approximate $(f \circ \arcsin)(x)$. $U_{\sin}$ can be constructed with n controlled-Ry rotations, and the unitary $\mathcal{U}^{\vec{\phi}}$ of Eq. (6) is slightly simpler to implement in

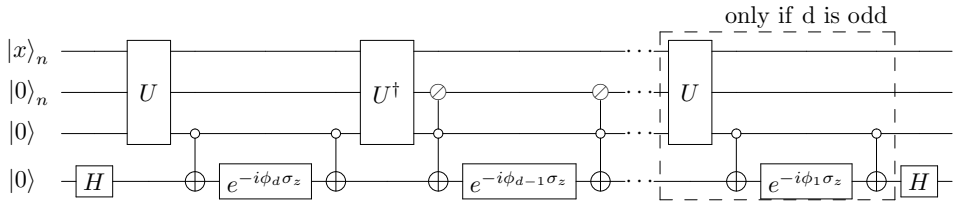

Figure 1: Circuit diagram of the $U^{\vec{\phi}}$ operator of Eq. (10) based on the unitary U of Eq. (4). The choice of angles $\vec{\phi} = (\phi_1, \phi_2, \cdots, \phi_d)$ determines the exact d-degree polynomial P applied to the state, such that the probability of measuring $|00\rangle$ in the bottom two qubits if d is odd and $|0\rangle_{n+2}$ in the bottom $n+2$ qubits if d is even, will be given by $|P(g(x))|^2$. We use the $\oslash$ symbol to indicate a unitary that is controlled on every qubit of a quantum register being in the $|0\rangle$ state.

this case because the corresponding projectors satisfy $\Pi = \tilde{\Pi} \equiv \mathbb{I}^{\otimes n} \otimes |0\rangle\langle 0|$. The choice of $U_{\sin}$ additionally simplifies the final state in Eq. (10), requiring n fewer ancilla qubits, leading to

$$U^{\vec{\phi}} |x\rangle_n |0\rangle_2 = |x_n\rangle \left(P(g(x)) |0\rangle_2 + \sqrt{1 - P(g(x))^2} |0_\perp\rangle_2 \right). \tag{12}$$

In this manuscript we instead consider the unitary

$$U_{\text{sqrt}} |x\rangle_n |0\rangle_{n+1} = |x\rangle_n \left(\sqrt{x} |\psi_0\rangle_n |0\rangle + \sqrt{1-x} |\psi_1\rangle_n |1\rangle \right), \tag{13}$$

such that QSP needs to approximate the function $f(x^2)$. The material difference between these two approaches lies mainly in the amount of resources required to construct $U_{\sin}$ and U_{sqrt}, as well as the polynomial degree needed to construct approximations of sufficient accuracy to the target transformation in each case. Because the details of QSP transformations based on $U_{\sin}$ are analyzed in Ref. [20], for the rest of this manuscript we focus on the implementation of the QSP transformation driven by U_{sqrt}, and revisit the former when we estimate the resources required to apply this technique in derivative pricing.

In this work we focus on target functions of the form $(Ae^x - B)/C$ because this particular form is ubiquitous in derivative pricing [21]. This motivation is further illustrated in practice in Sec. 5 where we apply these methods to price concrete financial derivatives. Namely, we are looking for a unitary which prepares the state

$$|x\rangle_n |0\rangle_{n+q} \to |x\rangle_n \left(\sqrt{\frac{Ae^x - B}{C}} |\psi_0'\rangle_n |0\rangle_q + \sqrt{1 - \frac{Ae^x - B}{C}} |\psi_1'\rangle_n |0_\perp\rangle_q \right), \tag{14}$$

for normalized quantum states $|\psi_0'\rangle_n$, $|\psi_1'\rangle_n$, and $|0_\perp\rangle_q$ denoting a normalized state orthogonal to $|0\rangle_q$. We assume that constants A, B, C are such that $(Ae^x - B)/C \in [0, 1]$.

4 Implementation

In this section we show how to implement the unitary U_{sqrt} of Eq. (13) and the choice of polynomial P needed in Eq. (10) in order to construct the target state Eq. (14) under various scenarios. We assume that the register $|x\rangle_n$ is an n-bit, binary, fixed-point representation of a scalar x with p digits to the left of the binary point

$$x = \underbrace{x_{n-1} \cdots x_{n-p}}_{p} . \underbrace{x_{n-p-1} \cdots x_0}_{n-p}. \tag{15}$$

Let us first consider the case where x is positive with $p = 0$, such that $x < 1$

$$x = \sum_{i=0}^{N-1} \frac{x_i \cdot 2^i}{N}, \tag{16}$$

where $x_i \in 0, 1$ denotes the i-th bit of $|x\rangle_n$ and $N = 2^n$. The first step to implement the transformation of Eq. (14) is to encode the value $\sqrt{x}$ to the amplitude of an ancilla qubit. We start with the state $|x\rangle_n$ and add n ancilla qubits in a uniform superposition

$$|x\rangle_n |+\rangle_n = \sum_{j=0}^{N-1} \frac{1}{\sqrt{N}} |x\rangle_n |j\rangle_n . \tag{17}$$

We then apply the unitary

$$\mathcal{C} : |a\rangle |b\rangle |0\rangle \to |a\rangle |b\rangle |a < b\rangle \tag{18}$$

to perform a binary comparison between the two qubit registers of Eq. (17). This produces the state

$$\sum_{j=0}^{N-1} \frac{1}{\sqrt{N}} |x\rangle_n |j\rangle_n |N \cdot x < j\rangle = |x\rangle_n \otimes \left[\sum_{j=0}^{N \cdot x - 1} \frac{1}{\sqrt{N}} |j\rangle_n |0\rangle + \sum_{j=N \cdot x}^{N-1} \frac{1}{\sqrt{N}} |j\rangle_n |1\rangle \right], \tag{19}$$

with x given by Eq. (16). By inspection, we observe that the probability of measuring $|0\rangle$ in the comparator ancilla qubit is then $\frac{N \cdot x}{N} = x$. The unitary $U_{\text{sqrt}} = \mathcal{C}(\mathbb{I}^{\otimes n} \otimes H^{\otimes n} \otimes \mathbb{I})$ acting on the state $|x\rangle_n |0\rangle_{n+1}$therefore creates the target state of Eq. (13)

$$U_{\text{sqrt}} |x\rangle_n |0\rangle_{n+1} = |x\rangle_n \left(\sqrt{x} |\psi_0\rangle_n |0\rangle + \sqrt{1 - x} |\psi_1\rangle_n |1\rangle \right), \tag{20}$$

where $|\psi_0\rangle_n$ and $|\psi_1\rangle_n$ are normalized quantum states which depend on the value of x. A polynomial P which approximates the function

$$f(x) = \sqrt{\frac{Ae^{x^2} - B}{C}}, \tag{21}$$

in the domain $x \in [0, 1]$ allows us to approximately prepare the target state given by Eq. (14) using Eq. (10).

When $p > 0$ such that $x \geq 1$, we can use the same unitary U_{sqrt} as above, ignoring the location of the binary point p, which in this case generates the state

$$U_{\text{sqrt}} |x\rangle_n |0\rangle_{n+1} = |x\rangle_n \left(\sqrt{\frac{x}{2^p}} |\psi_0\rangle_n |0\rangle + \sqrt{1 - \frac{x}{2^p}} |\psi_1\rangle_n |1\rangle \right). \tag{22}$$

Because we now have an extra factor of 2^p, the polynomial P in this case needs to approximate the function

$$f(x) = \sqrt{\frac{Ae^{x^2 \cdot 2^p} - B}{C}}, \tag{23}$$

in order to generate the target state of Eq. (14).

In order to handle negative values of x, we first need to pick a value $s \geq |x| \geq 0$ and then apply a binary addition circuit which performs $|x\rangle_n |0\rangle_m \to |x\rangle_n |x + s\rangle_m$, where $|x + s\rangle_m$ is again a fixed-point register with m total digits and p to the left of the binary point. The unitary U_{sqrt} of Eq. (22) is then applied with the second register as input to construct the state

$$U_{\text{sqrt}} |x + s\rangle_m |0\rangle_{m+1} = |x + s\rangle_m \left(\sqrt{\frac{x + s}{2^p}} |\psi_0\rangle_m |0\rangle + \sqrt{1 - \frac{x + s}{2^p}} |\psi_1\rangle_m |1\rangle \right), \tag{24}$$

where the normalization factor 2^p arising from treating the fixed-point register $|x + s\rangle_m$ as a binary string in the comparator circuit, similarly handles the case $x \leq -1.0$. The polynomial P which generates the target state of Eq. (14) needs to approximate the function

$$f(x) = \sqrt{\frac{Ae^{x^2 \cdot 2^p} e^{-s} - B}{C}}. \tag{25}$$

Note that for both positive and negative values of x, the transformation in Eq. (24) guarantees that the input to the polynomial P will only be the domain $x \in [0, 1]$. This way, while only polynomials with definite parity (either even or odd) can be implemented using QSP, we can extend the function we approximate with QSP to the domain $x \in [-1, 1]$ using $f(-x) = f(x)$ or $f(-x) = -f(x)$, even if $f(x)$ does not have definite parity. Additionally, the unitary U_{sqrt} of Eq. (24) and the function Eq. (25) allow us to construct the target state for any input x, provided that $f(x) \in [0, 1]$.

5 Applications

5.1 European Call Option

In this section we illustrate how the QSP method described in the previous sections to prepare the state in Eq. (14) can be used in derivative pricing to compute payoffs of the form $f(S) = \max(0, S - K)$, by first considering a European call option. Given a strike price K, the holder of a European call option contract receives the amount

$$f(S_T) = \max(0, S_T - K), \tag{26}$$

at a future date T (the *expiration date*), where S_T is the market price of an underlying asset at T. In order to determine the value of this contract today, the future price of the underlying asset is modeled as a stochastic variable with an appropriate probability distribution, and the price of the option is calculated by computing the expectation value of its *payoff* in Eq. (26).

The re-parameterization method for derivative pricing described in Ref. [8], implements the operator of Eq. (1) by loading a discretized normal distribution and then performing an affine transformation to the registers using quantum arithmetic to generate a superposition of possible asset log-returns r_i at time T, each weighed by the probability of occurrence

$$\sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n , \tag{27}$$

where $N = 2^n$. The price of the underlying asset for each possible log-return r_i is given by $S_i = S_0 e^{r_i}$, where S_0 is the price of the underlying asset today. The payoff operator of Eq. (2) is implemented by first calculating the price of the underlying asset from the log-return into another register

$$\sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n \rightarrow \sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n \left| S_i = S_0 e^{r_i} \right\rangle , \tag{28}$$

computing the payoff of Eq. (26)

$$\sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n \left| S_i \right\rangle \rightarrow \sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n \left| S_i \right\rangle \left| f(S_i) \right\rangle , \tag{29}$$

and finally encoding the payoff into the amplitude of an ancilla qubit

$$\begin{aligned} &\sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n \left| S_i \right\rangle \left| f(S_i) \right\rangle \rightarrow \\ &\qquad \sum_{i=0}^{N-1} \sqrt{p_i} \left| r_i \right\rangle_n \left| S_i \right\rangle \left| f(S_i) \right\rangle \left(\sqrt{\frac{\max(0, S_i - K)}{f_{\max}}} \left| 0 \right\rangle + \sqrt{\frac{1 - \max(0, S_i - K)}{f_{\max}}} \left| 1 \right\rangle \right), \end{aligned} \tag{30}$$

where $f_{\max} = e^{\max\{r_i\}} - K$ is the maximum value of the payoff possible by the discretization chosen in Eq. (27). The probability of measuring $|0\rangle$ in the last qubit is

$$\mathbb{P}[|0\rangle] = \sum_{i=0}^{N-1} p_i \frac{\max(0, S_i - K)}{f_{\max}}, \tag{31}$$

which is the expectation value of the option's payoff, normalized by $f_{\max}$. Thus, applying amplitude estimation to the last qubit allows us to estimate the price of the option.

We now describe how QSP and the methods developed in the previous sections can be used to perform the same computation. Let the $|r_i\rangle_n$ register of Eq. (27) represent each log-return in fixed-point representation and let $s = \ln(S_0/K)$. Apply a circuit which performs the comparison $|r_i\rangle_n |0\rangle \to |r_i\rangle_n |r_i \leq s\rangle$ to Eq. (27) to get

$$\sum_{\{i|r_i>s\}} \sqrt{p_i} |r_i\rangle_n |0\rangle + \sum_{\{i|r_i\leq s\}} \sqrt{p_i} |r_i\rangle_n |1\rangle . \tag{32}$$

Add an m-qubit register and compute $|r_i\rangle_n |0\rangle_m \to |r_i\rangle_n |r_i + |s|\rangle_m$, where m is large enough to hold the maximum value of $r_i + |s|$ in fixed-point representation, to get

$$\sum_{\{i|r_i>s\}} \sqrt{p_i} |r_i\rangle_n |r_i + |s|\rangle_m |0\rangle + \cdots , \tag{33}$$

where we ignore terms with the last qubit being in the $|1\rangle$ state. Note that $0 \leq r_i + |s| < 2^p$ if the register $|r_i + |s|\rangle_m$ has p digits to the left of the binary point. The application of the U_{sqrt} operator of Eq. (24) to the $|r_i + |s|\rangle_m$ register and a new qubit register $|0\rangle_{m+1}$ gives

$$\sum_{\{i|r_i>s\}} \sqrt{p_i} |r_i\rangle_n |r_i + |s|\rangle_m \left(\sqrt{\frac{r_i + |s|}{2^p}} |\psi_0\rangle_m |0\rangle + \sqrt{1 - \frac{r_i + |s|}{2^p}} |\psi_1\rangle_m |1\rangle \right) |0\rangle , \tag{34}$$

for normalized quantum states $|\psi_0\rangle$ and $|\psi_1\rangle$. Employing the QSP unitary $U_+^{\vec{\phi}}$ of Eq. (10) with phases $\vec{\phi} = (\phi_1, \phi_2, \cdots, \phi_d)$ chosen such that the polynomial P approximates the function

$$f(x) = \sqrt{\frac{S_0 e^{x^2 \cdot 2^p} e^{-|s|} - K}{f_{\max}}} \tag{35}$$

produces

$$\sum_{\{i|r_i>s\}} \sqrt{p_i} |r_i\rangle_n |r_i + |s|\rangle_m \left(\sqrt{\frac{S_0 e^{r_i} - K}{f_{\max}}} |\psi_0'\rangle_m |G\rangle \right) , \tag{36}$$

with $|G\rangle = |0\rangle_3$ for odd d, and

$$\sum_{\{i|r_i>s\}} \sqrt{p_i} |r_i\rangle_n |r_i + |s|\rangle_m \left(\sqrt{\frac{S_0 e^{r_i} - K}{f_{\max}}} |G\rangle \right) , \tag{37}$$

with $|G\rangle = |0\rangle_{m+3}$ for even d, where we have ignored terms with any qubit in the last register being in the $|1\rangle$ state. Applying amplitude estimation to estimate the probability of measuring the $|G\rangle$ state gives us

$$\mathbb{P}[|G\rangle] = \sum_{\{i|r_i>s\}} p_i \frac{S_0 e^{r_i} - K}{f_{\max}}, \tag{38}$$

which is the same as the value we estimate with the standard re-parameterization method in Eq. (31). In this case however, we do not explicitly compute the values $|S_i = S_0 e^{r_i}\rangle$ or $|f(S_i)\rangle$ using quantum arithmetic, but rather implicitly calculate them by applying amplitude transformations using QSP. In Fig. 2, we see plot the function of Eq. (35) with $S_0 = K = 100$, $p = 3$, $s = 0$ and $f_{\max} = S_0 e^{2^p} - K$ along with polynomial approximations of varying degrees constructed using the optimization-based method described in Appendix A.

We note that Ref. [20] discussed the possibility of applying QSP to compute common derivative payoffs such as that of Eq. (26). The authors pointed out that polynomial approximations to functions of the form $f(x) = \sqrt{x}$ suffer due to the fact that the function is not differentiable at the origin. By including the exponential in the function we are approximating, not only do we pass more computation to the QSP transformation, but we also address this issue since Eq. (35) is differentiable in the entire domain of applicability.

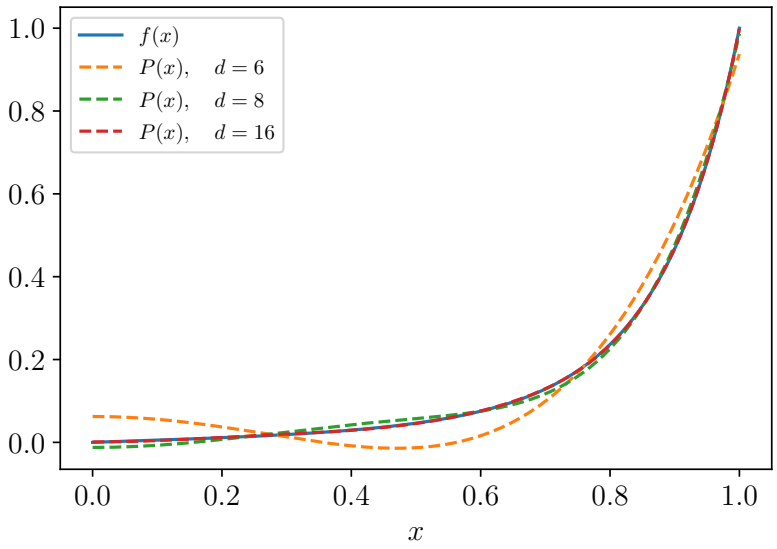

Figure 2: The function $f(x)$ of Eq. (35) with $S_0 = K = 100$, $p = 3$, $s = 0$ and $f_{\max} = S_0 e^{2^p} - K$ used to compute the payoff of a European call option, and approximating polynomials $P(x)$ of degree $d \in [6, 8, 16]$, which can be applied to a quantum system using the QSP unitary of Eq. (10). The polynomial approximations are generated using the method described in Appendix A.

5.2 Autocallable

For a single underlying asset, an *autocallable* derivative contract consists of

1. A set of k binary payoffs $\{K_i, t_i, f_i\}$, such that if the asset returns at time t_i exceed K_i, the holder of the contract receives a return f_i and the contract terminates. The asset's return at time t is defined as S_t/S_0, where S_t is the value of the asset at time t and S_0 the value of the asset today.

2. If none of the binary payoffs have been triggered, and the asset returns have crossed a barrier value B at any time before the contract's expiration, the holder receives a return given by $1 - \max(0, K_T - f_T)$, where $K_T \in [0, 1]$ is a pre-determined *strike* return and f_T is the asset's return at expiration date T.

Because an autocallable is typically defined in terms of an asset's performance, it also includes a (dollar) notional value V on which the payoff returns are evaluated to determine a dollar amount to be exchanged. For example, if the contract's notional value is \$1M, and a binary payoff with $f_i = 1.5$ is triggered, the holder of the contract receives \$1.5M in total. We now describe how this contract can be priced by employing QSP at the appropriate place.

Starting again from the re-parameterization method from [8], we first initialize a superposition of probability-weighted possible paths for the asset's log-returns at each timestep

$$\sum_r \sqrt{p_r}\,|r\rangle \qquad \text{with} \qquad |r\rangle = \bigotimes_{t=1}^{T} |r_t\rangle\,, \tag{39}$$

where the asset's total return at timestep t is given by e^{r_t}. We add a k-qubit register $|s\rangle$ and on each qubit s_i we calculate whether the k-th binary payoff is triggered, by checking whether the condition $r_{t_i} > \ln K_{t_i}$ is satisfied and no previous binary payoffs have been triggered

$$|s\rangle = \bigotimes_{i=1}^{k} |s_i = ((r_{t_i} > \ln K_{t_i}) \cdot (\neg s_{i-1}))\rangle\,, \tag{40}$$

where the $\neg$ symbol denotes logical negation and $s_0 = 0$. Each binary variable s_i will thus be 1 if the asset return at time t_i exceeds the threshold amount K_i and none of the previous payoffs have

been triggered, hence indicating whether the payoff at t_i should be payed. To determine whether the payoff from the second autocallable condition is triggered, we must check if the barrier value B has been crossed at any timestep of the simulation t. This is achieved by first adding a $(T+1)$-qubit register $|b\rangle$ and calculating $b_t = r_t \leq \ln B$ in each qubit for $t \in [1, T]$, and then computing

$$b_{T+1} = \left(\neg \prod_{t=1}^{T} b_i \right) \cdot \prod_{i=1}^{k} (\neg s_i) \,, \tag{41}$$

such that $b_{T+1} = 1$ if $r_t > B$ for at least one value of t, and every qubit in $|s\rangle$ is zero. Therefore, $b_{T+1} = 1$ indicates that the barrier B has been crossed at least once before the contract's expiration, and no binary payoffs have been triggered, meaning the second condition of the autocallable payoff is satisfied. Note that all these logical operations above can be computed using just CNOT and Toffoli gates.

Un-computing and dropping the first T qubits of the $|b\rangle$ register, we arrive at the state

$$\sum_r \sqrt{p_r} \, |r\rangle \bigotimes_{i=1}^{k} |s_i\rangle \, |b\rangle \,. \tag{42}$$

The $\mathcal{F}$ operator needed to encode the autocallable contract's payoff into the amplitude of an ancilla qubit as outlined in Eq. (2), should construct the state (up to normalization)

$$\sum_r \sqrt{p_r} \, |r\rangle \bigotimes_{i=1}^{k} |s_i\rangle \, |b\rangle \begin{cases} \left(\sqrt{f_i} \, |0\rangle + \sqrt{1 - f_i} \, |1\rangle\right) , & \text{if } s_i = 1 \\ \left(\sqrt{1 - \max(0, K_T - f_T)} \, |0\rangle + \sqrt{\max(0, K_T - f_T)} \, |1\rangle\right) , & \text{if } b = 1 \\ |1\rangle \,, & \text{otherwise} \end{cases} \tag{43}$$

where $f_T = e^{r_T}$ is the asset's return at the final timestep of the computation. The first payoff clause can be implemented using controlled Ry rotations since the values f_i are known in advance of the computation. For the second clause, we first apply a comparator circuit with an ancilla qubit to calculate $|r_T < \ln K_T\rangle$, and then controlled on this qubit, we need to prepare the state

$$|r_T\rangle_n \, |0\rangle_{n+q} \to |r_T\rangle_n \left(\sqrt{1 - (K_T - e^{r_T})} \, |\psi_0\rangle_n \, |0\rangle_q + \sqrt{(K_T - e^{r_T})} \, |\psi_1\rangle_n \, |0_\perp\rangle_q \right) , \tag{44}$$

where $|\psi_0\rangle_n$, $|\psi_1\rangle_n$ are arbitrary normalized states. Note that we have already imposed the condition $1 - (K_T - e^{r_T}) \in [0, 1]$, so the state is appropriately normalized. Since this transformation has the same form as Eq. (14), we can approximately prepare it using the QSP method described in Sections 3 and 4, similarly to the European call option described in Sec. 5.1. Assuming $|r_T\rangle_n$ is an n-qubit fixed-point representation of r_T with p digits to the left of the binary point and spans over both positive and negative values, we employ the unitary of Eq. (24) with an appropriately chosen value $s \geq \max(|r_T|) \geq 0$, to first generate the state

$$U_{\text{sqrt}} \, |r_T + s\rangle_m \, |0\rangle_{m+1} = |r_T + s\rangle_m \left(\sqrt{\frac{r_T + s}{2^p}} \, |\psi_0\rangle_m \, |0\rangle + \sqrt{1 - \frac{r_T + s}{2^p}} \, |\psi_1\rangle_m \, |0_\perp\rangle \right) . \tag{45}$$

Subsequently, the polynomial transformation P we apply with QSP in order to generate Eq. (44) approximates the function

$$f(x) = \sqrt{1 - (K_T - e^{x^2 \cdot 2^p} e^{-s})}. \tag{46}$$

While we have described an autocallable contract defined only on a single underlying asset, the same approach can be extended to multi-asset cases such as *BestOf* and *WorstOf* variants. In those cases, we generate stochastic paths for all underlying assets and apply a circuit to compare the relative performance of all assets, identifying the best/worst performing asset. The payoff is then implemented similarly to the single-asset case following Eq. (43), based on the returns of the identified asset.

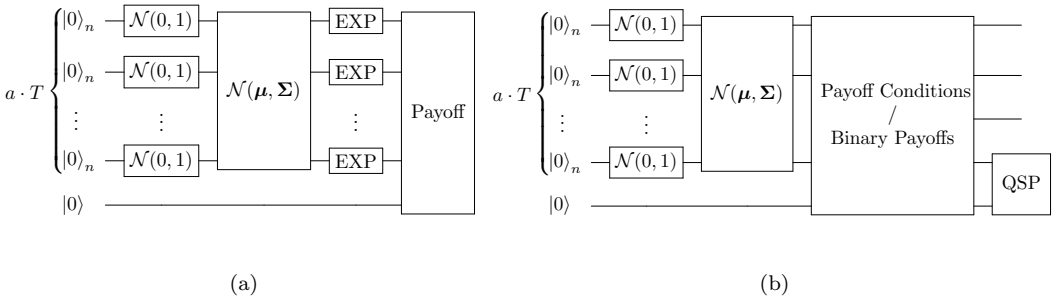

Figure 3: Schematic representation of the quantum circuits used to price the autocallable derivative contract using **(a)** the standard reparameterization method of Ref. [8] and **(b)** the QSP approach described in Sec. 5.2. The two approaches employ the same recipe to prepare the multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for each timestep $t \in [1, T]$ across d underlying assets. The application of QSP in the evaluation of the payoff via Eq. (43) removes the necessity of directly computing exponentials of quantum registers, a major contributor to the quantum resources required to implement the circuit [8]. The quantum circuits depict only the main quantum registers involved in the calculation and not ancillary registers that may be required in the subcomponents.

6 Resource Estimates for Quantum Advantage

We evaluate the efficiency of employing QSP in the evaluation of derivative payoffs by estimating the resources needed to price an autocallable derivative contract using the method described in Sec. 5.2. As a benchmark we use the resource estimates from Ref. [9], based on the reparameterization method introduced in Ref. [8] where coherent quantum arithmetic is used to compute the derivative payoff.

The goal of the reparameterization method is to prepare a probability distribution across a stochastic underlying assets and T timesteps under the assumption that asset prices follow geometric Brownian motion. In this setting, asset prices are described by a multivariate lognormal distribution, or equivalently, the asset log-returns are distributed normally. The reparameterization method prepares the distribution over log-returns by first loading independent standard normal distributions $\mathcal{N}(0,1)$ in $a \cdot T$ quantum registers, and then applying quantum arithmetic across the quantum registers to generate the multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ representing the distribution of log-returns at each timestep. The parameters $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are typically determined by historical market values and characterize the stochastic process of each asset. The price of each asset at each timestep is then calculated by exponentiating the resulting quantum registers, and the payoff is computed into another register based on the asset prices, before encoding the result into the amplitude of an ancilla qubit.

In the QSP approach outlined in Sec. 5.2, we again employ the reparameterization method to prepare the distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ over the assets' log-returns. We subsequently evaluate the payoff conditions of Eq. (40) and Eq. (41) using ancilla qubits, and encode the binary payoffs to the payoff ancilla qubit controlled on the corresponding payoff condition qubits. Note that the above does not require direct knowledge of asset prices, and can all be performed using only the log-returns, thus we do not need to calculate exponentials of values stored in binary representation in quantum registers. At the final timestep, we use QSP to implement the second clause of Eq. (43) which encodes the payoff directly in the amplitude of the payoff qubit, and as such we eliminate all instances of quantum register exponentiation in the circuit. A schematic representation of the quantum circuits implementing the standard reparameterization method of Ref. [8] and the corresponding QSP approach outlined in Sec. 5.2 in order to price an autocallable contract is shown In Fig. 3.

The resource estimates in Ref. [8, 9] for quantum advantage in pricing an autocallable contract are computed for an instance with $a = 3$ assets and $T = 20$ timesteps, and the overall target approximation error in the estimate is set to $\epsilon = 2 \times 10^{-3}$. The corresponding accuracy with which the payoff must be calculated and encoded into the amplitude of an ancilla qubit is $\epsilon_p = 10^{-3}$, meaning that the unitary $\mathcal{F}$ of Eq. (2) must be implemented such that the amplitude of the $|0\rangle$ is accurate to within ϵ_p. In the standard reparameterization circuit of Fig. 3a, this target

accuracy dictates the resources required to compute the exponentials, the quantum arithmetic in the computation of the payoff and the final rotation into an ancilla qubit using controlled-Ry gates. By virtue of Eq. (44), the QSP process encompasses the computation of the exponential and the payoff simultaneously, as well as the amplitude encoding once we have access to the unitary of Eq. (45) which encodes the value $\sqrt{r_T}$ in the amplitude of an ancilla qubit. The corresponding payoff resources of the QSP circuit in Fig. 3b are determined by the accuracy of the transformation of Eq. (44), which in turn depends on the degree d of the polynomial $P(x)$ required to approximate the function $f(x)$ of Eq. (46) such that $\max|f(x) - P(x)| \leq \epsilon_p = 10^{-3}$.

We extend the Q# [25] implementation from Ref. [9] to construct the quantum circuit of Fig. 3b and estimate the corresponding quantum resources when QSP is used to compute the exponentials and derivative payoff, for both $U_{\sin}$ (Eq. (11)) and U_{sqrt} (Eq. (13)) as the underlying unitary. In the original implementation, the parameters of the quantum circuit up to and including the block labeled $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ were chosen to satisfy the target accuracy requirements and are thus kept the same. As such, the register $|r_T\rangle$ uses the fixed-point representation of Eq. (15) with $(n, p) = (15, 5)$ such that $r_T \in [-R, R]$ with $R = 2^5$. Choosing $s = R$ in Eq. (45) makes sure the encoded amplitudes are positive, and the U_{sqrt} operator is constructed using the method described in Sec. 4 with a ripple-carry comparator circuit [26]. Using the polynomial approximation construction described in Appendix A, we numerically determine that a polynomial degree $d = 20$ suffices to generate a polynomial $P(x)$ which satisfies $\max|f(x) - P(x)| \leq 10^{-3}$ for all values of $K_T \in [0, 1]$. With these parameters, in Fig. 4, we plot the function $f(x)$ and the generated polynomial approximation $P(x)$, as well as the corresponding approximation error $|f(x) - P(x)|$ for $K_T = 1$.

The resource estimation from Ref. [8] was carried out with the assumption that piecewise polynomial approximation [27] is used to compute the exponentials required in the circuit. In Ref. [9], the quantum circuits were explicitly constructed, increasing the accuracy of the estimates, and the exponentials were computed using a lookup table. The lookup table is parameterized by the number of swap bits, offering tradeoffs between the circuit depth/width and the number of qubits required. The quantum resources reported are that of the $\mathcal{Q}$ operator driving amplitude estimation, where $\mathcal{Q} = \mathcal{A}S_0\mathcal{A}^\dagger S_1$, for appropriate reflection operators S_0 and S_1 [1], and the $\mathcal{A}$ operator of Eq. (3). In Table 1, we show the T-depth, T-count and number of logical qubits required to implement the $\mathcal{Q}$ operator with **(a)** the approach from Ref. [9] where a lookup table with one swap qubit is used to compute the exponentials, while arithmetic is used to calculate the payoff, and **(b)** the approach developed in this work where the exponentials and payoff are computed using QSP either with the unitary of Eq. (11) or the unitary of Eq. (13). We notice that the QSP method based on either unitary significantly outperforms the Arithmetic approach, but because the $U_{\sin}$-based QSP includes $d \cdot n$ total controlled-Ry rotations (n controlled-Ry rotations for each invocation of $U_{\sin}$ and d total invocations of $U_{\sin}$ and $U_{\sin}^\dagger$ in the QSP circuit), the total T-depth is significantly larger compared to using U_{sqrt}. In the Arithmetic column we use the results from Ref. [9] with one swap qubit for reference as it gives the most balanced resources for the metrics considered, but note that the U_{sqrt}-based QSP method outperforms the arithmetic method also in the case where five or ten swap qubits are used for the lookup table across all three metrics.

The difference in T-depth between the two QSP approaches can be understood by considering the Clifford+T decomposition of both $U_{\sin}$ and U_{sqrt}. The unitary U_{sqrt} can be implemented with a Toffoli-depth of $2\lfloor\log_2(n-1)\rfloor + 5$ using a ripple-carry comparator for an n-qubit register [26], which we can also regard as the equivalent T-depth if we use an ancilla qubit for the T-gate decomposition of each Toffoli [28]. Thus, U_{sqrt} can be constructed with a T-depth of 11 for the $n = 15$ case we consider here. On the other hand, the $U_{\sin}$ unitary requires Ry rotations, each of which can be performed to precision ϵ_R with a circuit of T-depth $3\log_2(1/\epsilon_R)$ [29]. Since there are $d \cdot n$ total Ry rotations in the QSP circuit, if we want the total error from all rotations to be within our total target payoff error $\epsilon_t \leq \epsilon_p = 10^{-3}$, each Ry rotation needs to be performed with precision $\epsilon_R = \epsilon_t/(d \cdot n) \sim 3 \times 10^{-6}$. Hence, each $U_{\sin}$ unitary requires a T-depth of $15 \cdot 3\log_2(1/\epsilon_R) \sim 818$, about 74 times larger than that of U_{sqrt}.

Using the Iterative Quantum Amplitude Estimation method [13] to compute the total resources needed for the end-to-end process to estimate the target amplitude to within $\epsilon = 10^{-3}$ at confidence level $1 - \alpha = 0.68$ [8], we find that the U_{sqrt}-QSP method requires 4.7k logical qubits, a total T-depth of 4.5×10^7 and T-count of 2.4×10^9. With the assumption that this derivative contract can be priced classically in one second [8], a quantum processor would need to execute T-gates at a

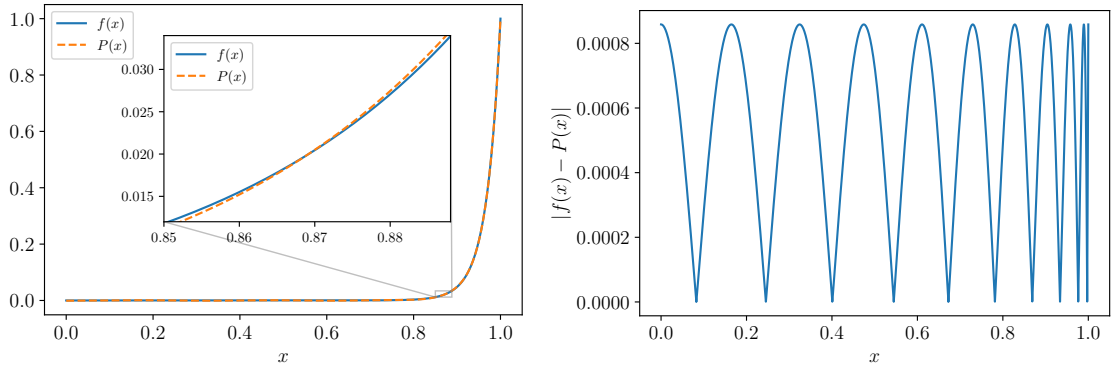

Figure 4: **(a)** The function $f(x)$ of Eq. (46) with $p = 5$, $s = 2^p$ and $K_T = 1$ used to compute the payoff of an autocallable contract, and a $d = 20$-degree approximating Chebyshev polynomial $P(x)$ generated using the method described in Appendix A. The QSP transformation of Eq. (10) based on the unitary U_{sqrt} of Eq. (45) can then be employed to approximately create the state in Eq. (44), encoding the autocallable payoff directly in a quantum amplitude. **(b)** The absolute error between $f(x)$ and the approximation $P(x)$ satisfies the target accuracy $\max |f(x) - P(x)| \leq 10^{-3}$.

Resource	**Arithmetic** [9]	**QSP (this work)**		**Improvement**
		U_{sin} [Eq. (11)]	U_{sqrt} [Eq. (13)]	
T**-depth**	36k	75k	7.8k	4.7x
T**-count**	6.6M	605k	414k	16x
# Qubits	19.2k	4.7k	4.7k	4.1x

Table 1: Quantum resources required to implement the $\mathcal{Q}$ operator driving Amplitude Estimation to price the autocallable contract from Sec. 6 with $a = 3$ assets and $T = 20$ timesteps. The Arithmetic column refers to the results from Ref. [9] where the exponentials are computed using a lookup table with one swap qubit, and the payoff using quantum arithmetic [8]. In the QSP column, the corresponding resources are shown when QSP is employed to compute both the exponentials and the payoff as described in Sec. 5.2, based on either Eq. (11) or Eq. (13) as the underlying unitary. In the last column we highlight the improvement of U_{sqrt}-QSP for each resource compared to the Arithmetic column. For all cases, the resources are calculated such that all encoded amplitudes are accurate to within $\epsilon_p \leq 10^{-3}$.

rate of 45MHz to match the classical performance if QSP with U_{sqrt} is used, compared to 207MHz with the arithmetic method and 430MHz using QSP and U_{sin}.

7 Discussion

It has been previously highlighted that quantum arithmetic is a major bottleneck in pricing financial derivatives on quantum computers [8, 9], and the QSP-based method we have introduced to construct derivative payoffs significantly reduces the quantum resources required for quantum advantage compared to previous methods, in both circuit width and depth. This approach seems more natural in the following sense: When an application requires the encoding of a function of a variable into a quantum amplitude, it should be more efficient to directly transform the amplitude, rather first applying a digital transformation through arithmetic on computational basis states and subsequently encoding the result into the amplitude. While QSP restricts the domain and range of transformations to the interval $[0, 1]$, we observe that in cases like the one studied in this manuscript it is still advantageous compared to utilizing additional quantum resources to increase the range/precision of digital operations. We are therefore hopeful that similar techniques can be explored in other use cases, potentially leading to additional improvements to the method. While the topic we are addressing in this manuscript is quite different from that of Ref. [20], the unitary

we introduced in Eq. (13) can also be employed for the state preparation method introduced in that work, potentially further reducing the resources in that case as well. Additionally, the availability of open source implementations for QSP phase factor calculations [30, 31] enables the exploration of QSP for specific applications more accessible, without requiring deep expertise in the topic.

One limitation of the QSP method in its current form is that it can only provide approximations to functions of a single variable. While for the applications considered in this work this is enough to provide an advantage, the ability to handle multivariate functions would greatly increase its applicability. For example, multivariable QSP (M-QSP) could potentially reduce the quantum resources in derivative pricing further by replacing the quantum arithmetic in the circuit component labeled $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ in Fig. 3. Moreover, it would enable the computation of more complicated derivative payoffs which are non-trivial functions of multiple stochastic variables. Some properties and aspects of M-QSP have been discussed in Ref. [32], and additional research on this topic would have potentially far-reaching practical consequences for quantum applications. We believe that this is a very worthwhile topic for further research.

8 Acknowledgments

We thank Rajiv Krishnakumar, Dave Clader and Farrokh Labib for useful discussions on derivative pricing and QSP, and Lin Lin for pointing us to QSPPACK to numerically generate polynomial approximations and phase factors used in QSP.

References

[1] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, "*Quantum Amplitude Amplification and Estimation*," Contemporary Mathematics **305** (2002).

[2] A. Montanaro, "*Quantum speedup of Monte Carlo methods*," Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences **471** (2015).

[3] P. Rebentrost, B. Gupt, and T. R. Bromley, "*Quantum computational finance: Monte Carlo pricing of financial derivatives*," Phys. Rev. A **98**, 022321 (2018).

[4] S. Woerner and D. J. Egger, "*Quantum risk analysis*," npj Quantum Information **5** (2019).

[5] 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).

[6] J. a. F. Doriguello, A. Luongo, J. Bao, P. Rebentrost, and M. Santha, "*Quantum Algorithm for Stochastic Optimal Stopping Problems with Applications in Finance*," in 17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022), Leibniz International Proceedings in Informatics (LIPIcs), Vol. 232 (Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2022) pp. 2:1–2:24.

[7] S. Herbert, "*Quantum Monte Carlo Integration: The Full Advantage in Minimal Circuit Depth*," Quantum **6**, 823 (2022).

[8] 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).

[9] "*Using Q# to estimate resources needed for quantum advantage in derivative pricing*," Accessed: 2023-06-21.

[10] C. Zoufal, A. Lucchi, and S. Woerner, "*Quantum Generative Adversarial Networks for learning and loading random distributions*," npj Quantum Information **5** (2019).

[11] 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).

[12] Y. Suzuki, S. Uno, R. Raymond, T. Tanaka, T. Onodera, and N. Yamamoto, "*Amplitude estimation without phase estimation*," Quantum Information Processing **19**, 75 (2020).

[13] D. Grinko, J. Gacon, C. Zoufal, and S. Woerner, "*Iterative quantum amplitude estimation*," npj Quantum Information **7** (2021).

[14] 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.

[15] J. M. Martyn, Z. M. Rossi, A. K. Tan, and I. L. Chuang, "*Grand Unification of Quantum Algorithms*," PRX Quantum **2** (2021).
[16] G. H. Low and I. L. Chuang, "*Optimal Hamiltonian Simulation by Quantum Signal Processing*," Phys. Rev. Lett. **118**, 010501 (2017).
[17] J. M. Martyn, Y. Liu, Z. E. Chin, and I. L. Chuang, "*Efficient fully-coherent quantum signal processing algorithms for real-time dynamics simulation*," The Journal of Chemical Physics **158**, 024106 (2023).
[18] L. Lin and Y. Tong, "*Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems*," Quantum **4**, 361 (2020).
[19] P. Rall and B. Fuller, "*Amplitude Estimation from Quantum Signal Processing*," Quantum **7**, 937 (2023).
[20] S. McArdle, A. Gilyén, and M. Berta, "*Quantum state preparation without coherent arithmetic*," arXiv preprint arXiv:2210.14892 (2022).
[21] J. Hull, *Options, futures, and other derivatives*, 6th ed. (Pearson Prentice Hall, Upper Saddle River, NJ [u.a.], 2006).
[22] J. Haah, "*Product Decomposition of Periodic Functions in Quantum Signal Processing*," Quantum **3**, 190 (2019).
[23] 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] .
[24] Y. Dong, X. Meng, K. B. Whaley, and L. Lin, "*Efficient phase-factor evaluation in quantum signal processing*," Physical Review A **103**, 042419 (2021).
[25] Microsoft, *Q# Language Specification* (2020).
[26] T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore, "*A logarithmic-depth quantum carry-lookahead adder*," Quantum Information and Computation **6**, 351 (2006).
[27] T. Häner, M. Roetteler, and K. M. Svore, "*Optimizing quantum circuits for arithmetic*," arXiv preprint arXiv:1805.12445 (2018).
[28] P. Selinger, "*Quantum circuits of T-depth one*," Physical Review A **87** (2013).
[29] N. J. Ross and P. Selinger, "*Optimal Ancilla-Free Clifford+T Approximation of z-Rotations*," Quantum Info. Comput. **16**, 901 (2016).
[30] "*QSPPACK*," Accessed: 2023-06-21.
[31] "*pyqsp*," Accessed: 2023-06-21.
[32] Z. M. Rossi and I. L. Chuang, "*Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle*," Quantum **6**, 811 (2022).
[33] 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).

A Construction of Polynomial Approximations for QSP

In order to approximately apply a singular value transformation given by a function $f(x)$ using QSP, we first need to find a polynomial which approximates it. The polynomial transformations possible through the QSP framework must have definite parity (even or odd), but because we only aim to transform real positive amplitudes created by the unitary of Eq. (24), we restrict our attention to the interval $[0, 1]$ due to symmetry.

We employ the optimization-based method described in Ref. [33] 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 $\{c_k\}$

$$P(x) = \sum_{k=0}^{d/2} c_k T_{2k}(x), \tag{47}$$

and aims to find the coefficients $\{c_k\}$ which give the best approximation to $f(x)$. 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 $\{c_k\}$

$$\begin{aligned} &\min_{\{c_k\}} \quad \left\{\max_{x_j\in[0,1]} |f(x_j) - P(x_j)|\right\} \\ &\text{s.t.} \quad P(x_j) = \textstyle\sum_k c_k A_{jk}, \quad |P(x_j)| \leq c, \quad \text{for } j = 0, 1, \ldots, M-1, \end{aligned} \tag{48}$$

where c is chosen close to 1 but preferably slightly smaller to avoid overshooting, and A_{jk} is a matrix of coefficients defined as $A_{jk} = T_{2k}(x_j)$ for $k = 0, 1, \ldots, d/2$.