Option Pricing in Classiq using Quantum Amplitude Estimation

Viraj Dsouza
Plaksha University, Punjab, India,
viraj.dsouza@plaksha.edu.in

Ron Cohen
Classiq Technologies, Tel Aviv, Israel,
ron@classiq.io

***Abstract*—We present an implementation of an option pricing model on the Classiq platform using Quantum Amplitude Estimation (QAE) algorithm, known to offer a quantum advantage in pricing complex options over classical Monte Carlo methods. Simulating the model without a limit on circuit width would result in very high qubit requirements, out of reach of currently available simulators. Using Classiq, we demonstrate results of simulations of the model for the limiting condition of circuit width and also show a possibility of depth optimization while keeping the limit on the circuit width. For executing the algorithm on available simulators, the circuit width was capped at 30 qubits. The insights obtained could be useful for learning about how quantum algorithms like QAE can be used for potential finance applications and also contribute in further improvements in implementations of quantum algorithms.**

Introduction

In today's financial markets, substantial computational resources are dedicated to pricing and managing financial assets and derivatives. Derivatives, with payoffs tied to underlying asset prices, pose a challenge due to their stochastic nature, known as the pricing problem ([1]). While the Black-Scholes-Merton model ([2] and [3]) offers a solution for some derivatives, complexities arise with more complex contracts and asset dynamics. Traditionally, Monte Carlo methods have been pivotal in tackling the pricing problem, leveraging computational power to compute expectation values of interest.

However, the emergence of quantum computing offers the promise of algorithmic speedups across various domains. Grover's search algorithm ([4]), a pioneer in quantum computing, demonstrates quadratic speedups in searching unstructured databases, with extensions to optimization tasks. Particularly, the amplitude estimation algorithm shows potential for significantly accelerating expectation value estimations, a task commonly handled by Monte Carlo methods. The intersection of finance and quantum algorithms ([5] presents an opportunity for revolutionizing derivative pricing and risk management.

In the realm of derivative pricing, pioneering works have paved the way for leveraging quantum algorithms. Initially, in ([6]), authors introduced a promising methodology by employing the Quantum Amplitude Estimation (QAE) algorithm. This work laid the foundation for subsequent research endeavors, including ([7]), where algorithms tailored for various classes of options were developed, with a focus on error mitigation strategies. More recently, in ([9]), authors provided insights into resource requirements for achieving quantum advantage in derivative pricing.

Basic Review of Options

Financial options[8], a specific derivative instrument, are contracts granting the right to buy (*call option*) or sell (*put option*) a financial instrument at a predetermined price (called strike price, K) before or at expiration. In the time leading to the expiration, the price of the option changes due to the stochastic nature of the stock prices underlying it. In a European call option scenario, where the stock's price (S) underlying the option contract surpasses the predetermined target (K) before the expiry date, the contract buyer profits $S - K$ (*option payoff*) after deducting the premium paid for the contract, if the option contract is exercised on or before the expiry date. If the stock price remains below the target, the buyer faces a loss of the premium as the option contract would not be exercised. In this case the option payoff is zero. The goal of option pricing is to estimate the option payoff at its maturity date. Subsequently, this estimate is used to obtain the present fair value of the options contract.

Numerical pricing, often using Monte Carlo methods, involves modeling the underlying asset's price as random variables $\mathbf{X} = \{X_1, X_2, ..., X_N\}$ (N stands for number of assets in the contract), generating M number of price paths $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_M\}$ for each asset from the probability distribution P followed by X. Note that $\mathbf{X_i}$ is a column vector with N rows where each row indicates asset price in the price path i. Let the option's payoff on each price path i be denoted as $f(\mathbf{X}_i)$, the expectation value of the payoff $E_P[f(X)]$ as an average across all paths can be written as:

$$E_P[f(\mathbf{X})] = \sum_{i=1}^{M} p_i f(\mathbf{X}_i) \tag{1}$$

p_i is the probability of realizing the price path i. The final step is to discount the calculated expectation value to get the option's fair value. Classical Monte Carlo methods demand substantial computational resources for accurate option price estimates, especially for complex options. QAE algorithm is known to exhibit a theoretical quadratic speed-up compared to classical Monte Carlo methods.[7]

Options are categorized as path-independent or path-dependent and can involve a single asset or multiple assets. The payoff of path-independent options depend on the asset price at a single point in time, making prior asset prices irrelevant. In contrast, the payoff of path-dependent options also rely on the evolution and history of asset prices. Pricing path-independent options on a single asset is straightforward; while path-independent options on multiple assets are slightly more complex and they involve multiple expensive payoff calculations. Quantum computing is envisioned to have the most significant impact in pricing complex, path-dependent options ([9]).

In this article, section 1 will review the theory and gate based implementation of QAE algorithm which uses the QPE algorithm. Next in 2, a step by step theoretical implementation of option pricing methodology closely tied with the methodology in [7], for a path independent option with a single asset class will be provided. Section 3 and 4 will contain the main contribution of this article, with section 3 providing an overview of implementing this methodology using Classiq. The results of the algorithm and comparison of resources will be shown for options with 4 and 8 possible price paths, in section 4.

QPE and QAE Review

Quantum Phase Estimation (QPE) ([11]) has been the key ingredient for many quantum algorithms. Suppose a unitary operator U has an eigenvector $|\psi\rangle$ with eigenvalue $e^{2\pi i\phi}$ where ϕ is unknown, that is, $U|\psi\rangle = e^{2\pi i\phi}|\psi\rangle$. The goal of QPE is to estimate ϕ. To perform this estimation, we assume that we have available black boxes (also called *oracles*) capable of preparing the state $|\psi\rangle$ and performing the controlled-U^{2^j} (or CU^{2^j}) operation, for suitable non-negative integers j. The action of controlled version of the U operator on an arbitrary state, is given by,

$$\alpha|0\rangle|\psi\rangle + \beta|1\rangle|\psi\rangle \xrightarrow{CU} \alpha|0\rangle|\psi\rangle + e^{2\pi i\phi}\beta|1\rangle|\psi\rangle$$

QPE requires two register sets where first register contains n qubits which are in state $|0\rangle$. n depends on the precision required in the estimated phase. Suppose that the qubits in the first register are numbered from 0 to $n-1$. Second register stores $|\psi\rangle$ and has as many qubits necessary to store $|\psi\rangle$. We are given controlled U^{2^j} operators as black-box functions. The QPE metholdogy can be summarised as follows:

1) Apply Hadamard to first register which has n

qubits, all initialized to $|0\rangle$. We get

$$|\Psi_1\rangle = \frac{1}{2^{n/2}}\big(|0\rangle+|1\rangle\big)\otimes\cdots\otimes\big(|0\rangle+|1\rangle\big)\,|\psi\rangle\,.$$

2) Apply CU^{2^j} gate where qubit $n-j$ is the control for $j = 0,\ldots,n-1$ and the qubits representing the state $|\psi\rangle$ being the target. We get,

$$|\Psi_2\rangle = \frac{1}{2^{n/2}}\big(|0\rangle + e^{2\pi i\phi 2^{n-1}}\,|1\rangle\big)\otimes\cdots \otimes\big(|0\rangle + e^{2\pi i\phi 2^{1}}\,|1\rangle\big)\otimes\big(|0\rangle + e^{2\pi i\phi 2^{0}}\,|1\rangle\big)\,|\psi\rangle \tag{2}$$

After some simplification, this reduces to,

$$|\Psi_2\rangle = \frac{1}{2^{n/2}}\sum_{k=0}^{2^n-1} e^{2\pi ik\phi}\,|k\rangle\,|\psi\rangle \tag{3}$$

The phase $2\pi\phi$ is going to be such that ϕ is between zero and one, so we can write it as a decimal in *binary* notation as follows. We express ϕ exactly in n bits. That is,

$$\phi = 0.\phi_0\ldots\phi_{n-1} = \frac{\phi_0\ldots\phi_{n-1}}{N} = \frac{x}{2^n},$$

for an *integer* x. Each ϕ_i is either 0 or 1. The resulting state now is,

$$|\Psi_2\rangle = \frac{1}{2^{n/2}}\sum_{k=0}^{2^n-1} e^{\frac{2\pi ikx}{N}}\,|k\rangle\,|\psi\rangle\,.$$

3) The state of the first register is clearly of the form of Quantum Fourier transform (QFT) of state $|x\rangle$, hence applying Inverse QFT (or $QFT^\dagger$) to the first register, we exactly measure $|x\rangle\,|\psi\rangle = |\phi_0\ldots\phi_{n-1}\rangle\,|\psi\rangle$.

$$\frac{1}{2^{n/2}}\sum_{k=0}^{2^n-1} e^{\frac{2\pi ikx}{N}}\,|k\rangle\,|\psi\rangle \xrightarrow{QFT^\dagger\otimes I} |x\rangle\,|\psi\rangle\,.$$

Finally the required phase is just $2\pi\phi$ where $\phi = \frac{x}{2^n}$.

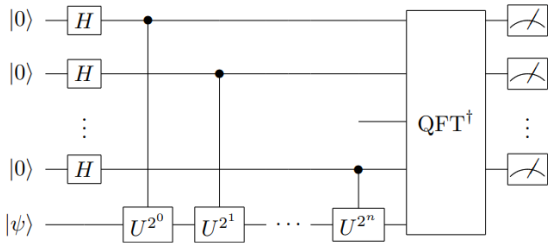


Figure 1: QPE circuit

For the case where ϕ cannot be written exactly with a n bit binary expression (i.e $e^{i2\pi\phi} \neq e^{i2\pi(\frac{x}{2^n})}$ for an integer x) the QPE algorithm would still produce a very good *approximation* of ϕ. In this case, $QFT^\dagger$ on 3 followed by simplification of the summation yields the state

$$\frac{1}{2^n}\sum_{k'}\left[\frac{1-\left[e^{\left(\frac{-2\pi i}{2^n}(k'-\phi 2^n)\right)}\right]^{2^n}}{1-\left[e^{\left(\frac{-2\pi i}{2^n}(k'-\phi 2^n)\right)}\right]}\right]|k'\rangle\,|\psi\rangle \tag{4}$$

Thus, probability of observing the state k' is,

$$P(k') = \frac{1}{2^{2n}}\left|\left[\frac{1-\left[e^{\left(\frac{-2\pi i}{2^n}(k'-\phi 2^n)\right)}\right]^{2^n}}{1-\left[e^{\left(-\frac{2\pi i}{2^n}(k'-\phi 2^n)\right)}\right]}\right]\right|^2 \tag{5}$$

The probability peaks when $k' \approx \phi 2^n$. Hence we have an approximate estimate of ϕ.
For example in the figure below n=3, $k' \approx 1$, hence estimated phase, $2\pi\phi \approx \frac{2\pi}{8} = 0.785$.

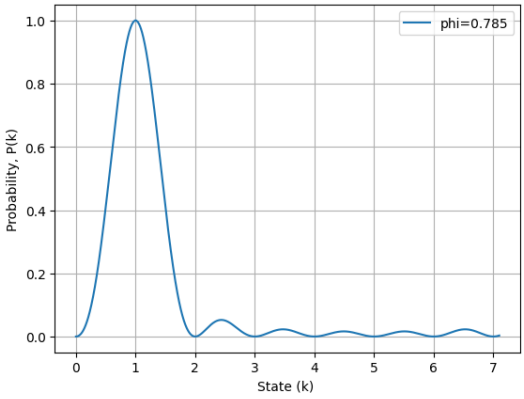


Figure 2: Example histogram based curve of QPE for single eigenstate

Phase Estimation for a superposition of eigenstates

For the case where $|\psi\rangle$ is a linear superposition of states $|\psi_p\rangle$, that is $|\psi\rangle = \sum_p c_p\,|\psi_p\rangle$ and the unitary operator U is such that $U\,|\psi_p\rangle = e^{\frac{2\pi i\phi_p}{2^n}}\,|\psi_p\rangle$. Now,

$$\begin{aligned} QPE(|0\rangle^{\otimes n}\,|\psi\rangle) &= \sum_p c_p QPE(|0\rangle^{\otimes n}\,|\psi_p\rangle) \\ &= \sum_p c_p(|\phi_p\rangle\,|\psi_p\rangle) \end{aligned} \tag{6}$$

Now projecting this state onto $|\phi_p\rangle\,|\psi_p\rangle$ gives the corresponding phase ϕ_p with probability $|c_p|^2$.

Here too we get a similar expression for probability P as function of state k' in the superposition as,

$$P(k') = \frac{1}{2^{2n}} \sum_p |c_p|^2 \left| \left[\frac{1 - \left[e^{\left(\frac{-2\pi i}{2^n} (k' - \phi_p 2^n) \right)} \right]^{2^n}}{1 - \left[e^{\left(-\frac{2\pi i}{2^n} (k' - \phi_p 2^n) \right)} \right]} \right] \right|^2 \tag{7}$$

The probability function now peaks when $k' \approx \phi_p 2^n$. Hence for this case we get multiple peaks corresponding to multiple phases in the superposition state. Due to the linear superposition, QPE on such states won't precisely give the phase corresponding to each state.

Quantum Amplitude Estimation (QAE)

QAE ([12]) uses QPE to estimate the amplitude of a *desired state* (or good state) in a quantum superposition. Let's say we have a unitary operator A acting on a register of $(n+1)$ qubits in the $|0\rangle$ state, resulting in the state $|\Psi\rangle$. Let $|\Psi\rangle$ be decomposed into a linear combination of n-qubit normalized states $|\psi_0\rangle_n$ (bad state) and $|\psi_1\rangle_n$ (good state) , where $a \in [0,1]$ is unknown. The goal of QAE is to estimate a.

$$A|0\rangle_{n+1} = |\Psi\rangle = \sqrt{1-a}\,|\psi_0\rangle_n |0\rangle + \sqrt{a}\,|\psi_1\rangle_n |1\rangle \tag{8}$$

We use the Grover Operator defined by:

$$Q = AS_0A^\dagger S_{|\psi_0\rangle} \tag{9}$$

where $S_0 = 1 - 2|0\rangle\langle 0|$ and $S_{|\psi_0\rangle} = 1 - 2|\psi_0\rangle|0\rangle\langle\psi_0|\langle 0|$, represent the reflection operators about the state $|0\rangle$ and the good state $|\psi_0\rangle$ respectively. The full action of the operator Q is to rotate the state $A|0\rangle_{n+1}$ by an angle 2θ in the $2D$ space spanned by $|\psi_0\rangle_n|0\rangle$ and $|\psi_1\rangle_n|1\rangle$, where $a = \sin^2(\theta)$.

The operator Q has the following matrix representation:

$$Q = \begin{pmatrix} \cos 2\theta & \sin 2\theta \\ -\sin 2\theta & \cos 2\theta \end{pmatrix} = (\cos 2\theta)1 + i(\sin 2\theta)\sigma_y \tag{10}$$

The eigenvalues of Q can be found to be $\exp(+i2\theta)$ and $\exp(-i2\theta)$. Let $|\psi_+\rangle$ and $|\psi_-\rangle$ represent the corresponding eigenvectors respectively. In the eigenbasis of Q we can express our state Ψ as,

$$\Psi = a_1|\psi_+\rangle + a_2|\psi_-\rangle$$

Now, the action of Q on $|\Psi\rangle$ is,

$$Q|\Psi\rangle = a_1 e^{+i2\theta}|\psi_+\rangle + a_2 e^{-i2\theta}|\psi_-\rangle$$

Let's denote $2\theta = \frac{2\pi q}{2^m}$, where m is the size of the estimation register in the QPE algorithm.

Now applying QPE using controlled Q operators,

$$QPE(|0\rangle|\Psi\rangle) = a_1|q\rangle|\psi_+\rangle + a_2|q + 2^{m-1}\rangle|\psi_-\rangle \tag{11}$$

Note that the phases corresponding to the eigenvectors $|\psi_+\rangle$ and $|\psi_-\rangle$ in 11 differ by angle π; hence the term $|q + 2^{m-1}\rangle$ in the above expression makes sense as,

$$\frac{2\pi(q + 2^{m-1})}{2^m} - \frac{2\pi q}{2^m} = \pi$$

If upon measurement of the estimation register we obtain $q' > 2^{m-1}$ then we get the phase corresponding to $|\psi_-\rangle$ branch, hence our estimated phase (say q_p)is,

$$q_p = q' - 2^{m-1}$$

Thus the value of 2θ is $\frac{2\pi q_p}{2^m}$ and hence,

$$a = \sin^2\theta = \sin^2\left(\frac{\pi q_p}{2^m}\right) \tag{12}$$

This result is exact, for the case where q_p is an integer. For the case where q_p is not an integer, QPE will yield an estimate for θ as seen earlier in the review for QPE.

Option Pricing Methodology

The article closely follows the option pricing methodology described in [7].
For an option with payoff f, let's suppose the $\mathcal{A}$ operator creates the following state:

$$\sum_{i=0}^{2^n-1} \sqrt{1 - f(S_i)}\sqrt{p_i}\,|S_i\rangle|0\rangle + \sum_{i=0}^{2^n-1} \sqrt{f(S_i)}\sqrt{p_i}\,|S_i\rangle|1\rangle.$$

where S_i represents the possible values the underlying asset can take and the corresponding probabilities are p_i. $f(S_i)$ represents the payoff if the price is S_i. Here we have assumed that the prices can take 2^n possible values on expiration and a quantum circuit n qubits can be used to represent all these price paths.
Comparing with equation 8, we find that

$$a = \sum_{i=0}^{2^n-1} f(S_i)\,p_i = E[f(S)]$$

The value of a which can be estimated using QAE.

These are the major steps involved:

1) Load the probability distribution P of the random variables $\mathbf{X}$
The price of a path-independent option depends only on the distribution of the underlying asset price S_t at the option maturity t and the payoff function f of the option. The distribution of S_t is truncated to the range $[S_{t,\,\min}, S_{t,\max}]$ and this interval is discretized to $\{0, \ldots, 2^n - 1\}$ to encode 2^n potential price values the asset can take upon expiration, utilizing n qubits. The distribution loading operator $\mathcal{P}$ creates the state:

$$|0\rangle_n \xrightarrow{\mathcal{P}} |\psi\rangle = \sum_{i=0}^{2^n-1} \sqrt{p_i}|i\rangle \tag{13}$$

with i to represent S_t. The n-qubit state $|i\rangle = |i_{n-1} \ldots i_0\rangle$ encodes the integer $i = 2^{n-1}i_{n-1} + \ldots + 2i_1 + i_0 \in \{0, \ldots, 2^n - 1\}$ with $i_k \in \{0, 1\}$ and $k = 0, \ldots, n-1$. The outcomes (S_t) of the random variable X has been mapped to the integer set $\{0, \ldots, 2^n - 1\}$ using an affine mapping:

$$S_t = S_{t,\min} + \frac{S_{t,\max} - S_{t,\min}}{2^n - 1} i$$

2) Loading the payoff $f(\mathbf{X})$ into the quantum circuit for all possible values of $\mathbf{X}$
An example for a call option payoff is the European Call Option $f(S_t)$ given as:

$$f(S_t) = \max(0, S_t - K)$$

To load this pay-off function into the quantum circuit and compute the expectation value of the pay-off, the state in 13 along with an ancilla qubit we create a quantum circuit that prepares the below state:

$$|i\rangle|0\rangle \to \sqrt{1 - \tilde{f}(i)}|i\rangle|0\rangle + \sqrt{\tilde{f}(i)}|i\rangle|1\rangle$$

Where $\tilde{f}$ is a normalized and rescaled function of the given f :

$$c\tilde{f}(i) = 2c\left(\frac{f(\phi(i)) - f_{\min}}{f_{\max} - f_{\min}}\right) - c \tag{14}$$

where

$$\phi(i) = S_{t,\min} + \frac{S_{t,\max} - S_{t,\min}}{2^n - 1} i$$

with $f_{\min} = \min_i f(i)$ and $f_{\max} = \max_i f(i)$, and $c \in [0, 1]$ is an additional scaling parameter. The choice of 14 is such that $c\tilde{f}(i) \in [-c, c]$. Thus, the following transformation can be achieved on the state $|\psi\rangle|0\rangle$ using controlled-Y rotations:

$$|\psi\rangle|0\rangle \to \sum_{i=0}^{2^n-1} \sqrt{p_i}|i\rangle \left[\cos\left(c\tilde{f}(i) + \frac{\pi}{4}\right)|0\rangle + \sin\left(c\tilde{f}(i) + \frac{\pi}{4}\right)|1\rangle\right] \tag{15}$$

3) Calculate the expectation value of the payoff $E_P[f(\mathbf{X})]$.
This is done by using QAE on 15 to calculate proabibility of ancilla qubit being in $|1\rangle$. The expression for the probability to find the ancilla qubit in state $|1\rangle$, can be calculated as follows,

$$P_1 = \sum_{i=0}^{2^n-1} p_i \sin^2\left(c\tilde{f}(i) + \frac{\pi}{4}\right)$$

For small values of $c\tilde{f}(i)$ (possible after an appropriate choice of c is made), the approximation $\sin^2\left(c\tilde{f}(i) + \frac{\pi}{4}\right) \approx c\tilde{f}(i) + \frac{1}{2}$, leads to

$$\begin{aligned} P_1 &\approx \sum_{i=0}^{2^n-1} p_i \left(c\tilde{f}(i) + \frac{1}{2}\right) \\ &= 2c\frac{E[f(X)] - f_{\min}}{f_{\max} - f_{\min}} - c + \frac{1}{2} \end{aligned} \tag{16}$$

Thus we could recover $E[\max(0, i - K)]$ from P_1 up to a scaling factor and a constant. Note that P_1 is computed through QAE algorithm on the state in 15.

Option Pricing using Classiq platform

After having gained familiarity with the theoretical implementation of the Option pricing model for European call options, this section would now provide for a practical implementation process of the theoretical steps, using the Classiq platform.

Brief overview of Classiq

In Classiq[10], quantum algorithms are constructed from functions, which are high-level building blocks that implement quantum logic and the data flow between them. Classiq SDK has several built-in functions that can be directly used or we can also build our own set of functional blocks from scratch. Each part of the algorithm (state preparation, loading payoff function as amplitudes, phase estimation) was designed by constructing such functional blocks.

The functional blocks are then wired together to form a quantum model and the next step is to

synthesize the model. The synthesis process constructs the quantum circuit as required by our model. The constraints like circuit depth, width, synthesis time etc and optimization parameters like width, depth can be specified if needed before synthesizing the quantum circuit. The platform optimizes the quantum program functions by storing multiple implementations for the same functionality. Such compilation approach allows for scaling of the problem in a given width limit. Execution of the synthesized circuit will then execute (or measure) the quantum circuit as required by our model. (number of shots, backend preferences and execution time can be defined).

Option Pricing Process

1) **Preparing all the functional blocks:**
We mainly require two functional blocks. The first functional block represents the algorithm $\mathcal{A}$; it involves loading the qubit states, $|S_i\rangle$ with relevant probabilities, p_i as in 13 and loading the scaled pay-off function as in 15. The states $|S_i\rangle$ represent the possible option prices at maturity. For the purpose of this paper, we work with two cases of a path-independent single-asset options, one that has 4 and the other 8 possible price paths with the following (arbitrary) distribution. We fix the strike price in both cases to be $K = 70$ (an arbitary choice).

   a) 4 **Price-Paths option:**
   Distribution of prices:

   $$p = [0.20, 0.35, 0.25, 0.20]$$

   Price-Paths:

   $$Y = 50[0.9, 1.18, 1.47, 1.75]$$

   where 50 is the initial price of the stock underlying the option.
   The classically computed expected-payoff using 1 turns out to be 4.35. Here $[S_{t,\,\text{min}}\,, S_{t,\text{max}}] = [45, 87.5]$

   b) 8 **Price-Paths option:**
   Distribution of prices:

   $$p = [0.04, 0.20, 0.17, 0.25, 0.08, 0.05, 0.18, 0.03]$$

   Price-Paths:

   $$S = 50[0.9, 1.1, 1.3, 1.5, 1.7, 1.9, 2.1, 2.3]$$

   where 50 is the initial price of the stock underlying the option.
   The classically computed expected-payoff using 1 turns out to be 11.35. Here $[S_{t,\,\text{min}}\,, S_{t,\text{max}}] = [45, 115]$

   We calculate expectation values of payoffs from the implementation of the algorithm with different estimation register sizes (m). The reason we chose a arbitrary distribution like above is to make the price distribution user-defined rather than drawing out samples from well-known price distributions (log-Normal, Exponential, etc). The latter can also be done using the exact same methodology.

   The state preparation results for both the above probability distributions have been provided in Appendix A. One can notice that the preparation is is almost exact and contain very small errors that gets carried forward to the next stages of the algorithm. The second functional block defines Grover's operator as in 9.

2) **Wiring the functional blocks and performing Quantum Phase Estimation:**
We use QPE algorithm using the Grover's operator for the iterative controlled operations in QPE (refer 1). QPE will be used on the state prepared by the first functional block. Measurement results of the estimation register will provide the state with maximum probability. In an ideal case, the probability distribution of the measurement results should have two peaks. If one peak is at state $|q\rangle$, the other would be at $|q + 2^{m-1}\rangle$ where m is the estimation register size. From either of the states, using 12 the scaled value of expectation can be obtained which is then rescaled appropriately using 16.

Results

The algorithm has been simulated using the NVIDIA GPU simulator available in Classiq, simulating up to 30 qubits. The algorithm was run for m values of $2, 3, 4$ and c (rescaling factor) values of $0, 5, 0.05, 0.005$ for both the options. Multiple cases where explored like not putting any limit on circuit width, putting a limit on circuit width and limiting circuit width while optimizing for depth. The circuits implemented in all of these cases are different, showing multiple implementations of the same algorithm but the expected pay-off results are exactly the same. A trade-off between resources used (width vs depth)

can be observed across the cases.

1) 4 **Price-Path Options:**

- **Case 1:** No width constraint
  The algorithm execution without the constraint on circuit width was done using Nvidia statevector simulato (this case is the default option of simulating the model). The simulation (5a) was executed for $c = 0.05$. For $m \geq 4$ circuit width turns out to be more than 30 and execution was not possible, suggesting that limiting the circuit width is a useful constraint to have for executing on available simulators. Table 1 contains the necessary results.
- **Case 2:** Circuit width capped at 30
  Table 2 provides for the pay-off results and the resources used for this case. The histogram results are shown in 3. Although for different c values the expected pay-off results are the same, the histograms have clearly seggregated double peaks as expected from theory for very low c's. Thus we see that, we can have the same results but using fewer qubits by putting a width limit on the synthesis using Classiq and get the same expected pay-off. This shows that multiple implementations for an algorithm is possible and we need to make the best choice based on the available resources.
- **Case 3:** Circuit width capped at 30 and depth optimization
  When depth optimization was carried out (keeping the limit on circuit width), the algorithm required higher number qubits but for this case (4 price paths) a reduction in circuit depth was not observed. This could be because of the time provided to the synthesis engine to optimize the circuit. In the provided time the engine could not look for better circuit depths but this is not the case when there are options with more price paths as we will see soon. The expectation results are the same for all cases, only the gate based implementation of the algorithm differ.

Table 1: No constraint on width and depth

	m	$E_P[f(X)]$	(Depth,Width)
$c = 0.05$	2	8.5	(760, 15)
	3	8.5	(1662, 28)
	4	–	(3464, 53)

Table 2: Circuit width capped at 30 (Case 2)

	m	$E_P[f(X)]$	(Depth,Width)
$c = 0.005$	2	8.5	(760, 6)
	3	8.5	(1662, 7)
	4	8.5	(3464, 8)
$c = 0.05$	2	8.5	(760, 6)
	3	8.5	(1662, 7)
	4	8.5	(3464, 8)
$c = 0.5$	2	8.5	(760, 6)
	3	8.5	(1662, 7)
	4	8.5	(3464, 8)

2) 8 **Price-Path Options:**
Table 4 contains the necessary results for the case where the circuit width wasn't limited. The histogram result is shown in 5b. For $m \geq 3$, the circuit width turns out to be more than 30 and execution was not possible. Table 5 and 6 provide results for the second and third cases, similar to the ones explained in the 4 price path case. The histogram results are shown in 4. A width vs depth trade-off can be clearly seen in the case of $8-$price-paths when depth optimization was performed keeping the width constant. This is a useful optimization (at the cost of some qubits) as circuit depth increases exponentially with m. We also see that the rescaling factor of $c = 0.5$ is not a good choice for this price distribution. Its useful to note from 16 that any choice of c won't work as P_1 represents probability which needs to be bounded between $[0, 1]$.

CONCLUSION

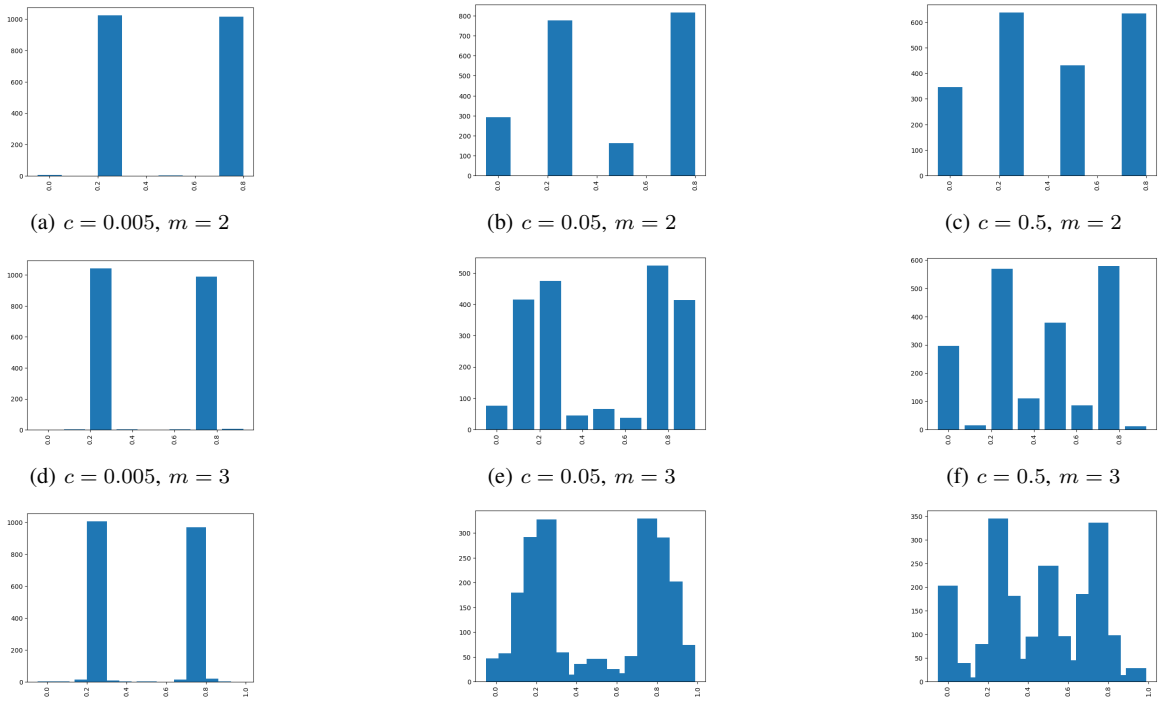
We have provided an implementation of Option Pricing methodology for European call options using Classiq platform through a high level functional

Table 3: With Depth Optimization(Case 3)

	m	$E_P[f(X)]$	(Depth,Width)
$c = 0.005$	2	8.5	(772, 21)
	3	8.5	(1690, 30)
	4	8.5	(3464, 8)

Table 4: No constraint on width and depth

	m	$E_P[f(X)]$	(Depth,Width)
$c = 0.05$	2	22.5	(2113, 30)
	3	–	(4559, 59)
	4	–	(9569, 116)

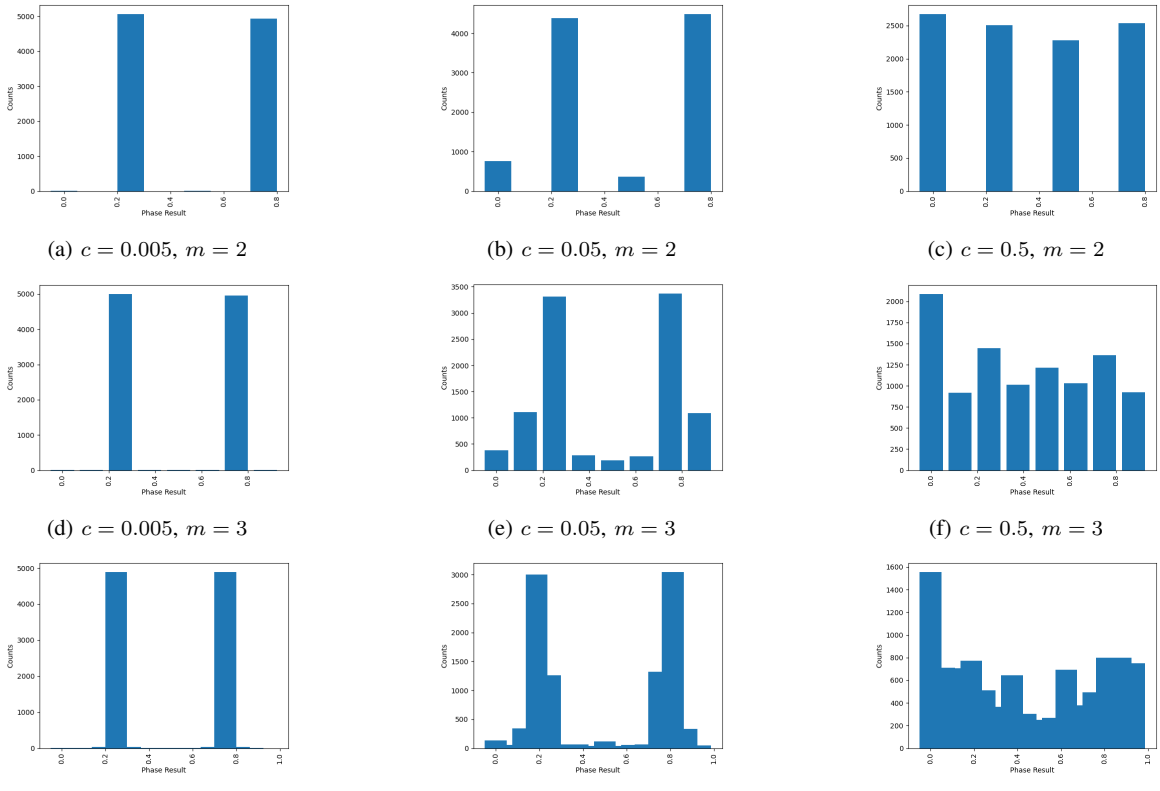


(a) $c = 0.005$, $m = 2$ (b) $c = 0.05$, $m = 2$ (c) $c = 0.5$, $m = 2$

(d) $c = 0.005$, $m = 3$ (e) $c = 0.05$, $m = 3$ (f) $c = 0.5$, $m = 3$

(g) $c = 0.005$, $m = 4$ (h) $c = 0.05$, $m = 4$ (i) $c = 0.5$, $m = 4$

Figure 3: **Execution results (shots vs Phase estimated) for Option with** 4 **Price paths**

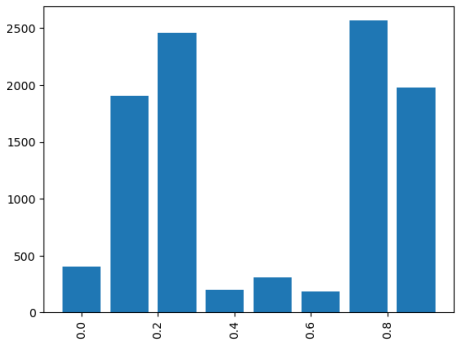


(a) $c = 0.005$, $m = 2$ (b) $c = 0.05$, $m = 2$ (c) $c = 0.5$, $m = 2$

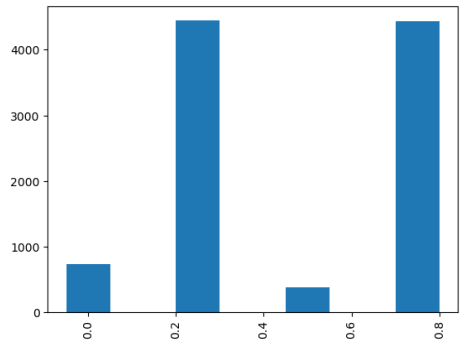
(d) $c = 0.005$, $m = 3$ (e) $c = 0.05$, $m = 3$ (f) $c = 0.5$, $m = 3$

(g) $c = 0.005$, $m = 4$ (h) $c = 0.05$, $m = 4$ (i) $c = 0.5$, $m = 4$

Figure 4: **Execution results (shots vs Phase estimated) for Option with** 8 **Price paths**



(a) $c = 0.05$, $m = 3$, width= 28



(b) $c = 0.05$, $m = 2$, width=30

Figure 5: **Execution results for Option with 4 and 8 Price paths (without circuit width constraint)**

Table 5: Circuit width capped at 30 (Case 2)

	m	$E_P[f(X)]$	(Depth,Width)
$c =$ 0.005	2	22.5	$(2114, 8)$
	3	22.5	$(4601, 9)$
	4	22.5	$(9572, 10)$
$c = 0.05$	2	22.5	$(2114, 8)$
	3	22.5	$(4601, 9)$
	4	22.5	$(9572, 10)$
$c = 0.5$	2	0	$(2114, 8)$
	3	0	$(4601, 9)$
	4	0	$(9572, 10)$

Table 6: With Depth Optimization(Case 3)

	m	$E_P[f(X)]$	(Depth,Width)
$c =$ 0.005	2	22.5	$(2035, 30)$
	3	22.5	$(4417, 30)$
	4	22.5	$(9572, 10)$

design. The results of expected pay-off and the resources used have been shown for the cases of 4 and 8 price paths. We have also demonstrated the opportunity to optimize circuit depth in the algorithm at the cost of more qubits; this is particularly useful as the depth of the algorithm increases exponentially with the size of the phase estimation register. Potential future directions could involve simulating for different types of options and even path-dependent options, using the algorithmic base explained in this article and also leveraging the approach used in a very recent work on rainbow options [13]. The scalability of the existing implementation could also be tested and the algorithm can also be executed on available noisy quantum hardware.

ACKNOWLEDGMENTS

We extend our gratitude to the Classiq Community, available via Slack, for their support in addressing platform-related queries and for providing timely suggestions. Additionally, we acknowledge the Qworld Association for instigating the project that culminated in this paper.

The codes are available in the project's public git repository found here and readers are welcome to try out the algorithm for higher number of price paths and explore further optimization in the depth by changing relevant parameters in the algorithm

REFERENCES

1. H. Follmer and A. Schied, Stochastic Finance: An Introduction in Discrete Time Published by Walter de Gruyter, 2004.
2. F. Black and M. Scholes, Journal of Political Economy 81, 637 (1973)
3. R. C. Merton, The Bell Journal of Economics and Management Science 4, 141 (1973).
4. L. K. Grover, in STOC '96 Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (ACM, New York, 1996), pp. 212219.
5. R. Or´us, S. Mugel, and E. Lizaso, "Quantum computing for finance: Overview and prospects," Reviews in Physics, vol. 4, p. 100028, Nov. 2019. [Online]. Available: https://doi.org/10.1016/j.revip.2019.100028
6. P. Rebentrost, B. Gupt, and T. R. Bromley, "Quantum computational finance: Monte carlo pricing of financial derivatives," Physical Review A, vol. 98, no. 2, Aug. 2018. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.98.022321
7. Nikitas Stamatopoulos, Daniel J. Egger, Yue Sun, Christa Zoufal, Raban Iten, Ning Shen, and Stefan Woerner, Option Pricing using Quantum Computers, Quantum 4, 291 (2020) [Online]. Available: http://dx.doi.org/10.22331/q-2020-07-06-291

8. John C. Hull, Options, futures, and other derivatives, 6th ed. (Pearson Prentice Hall, Upper Saddle River, NJ [u.a.], 2006).
9. S. Chakrabarti, R. Krishnakumar, G. Mazzola, N. Stamatopoulos, S. Woerner, and W. J. Zeng, “A threshold for quantum advantage in derivative pricing,” Quantum, vol. 5, p. 463, Jun. 2021. [Online]. Available: http://dx.doi.org/10.22331/q-2021-06-01-463
10. Classiq Technologies Ltd, “Classiq platform.” [Online]. Available: https://platform.classiq.io
11. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition, 10th ed. USA: Cambridge University Press, 2011.
12. Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp, “Quantum Amplitude Amplification and Estimation,” Contemporary Mathematics 305 (2002), 10.1090/conm/305/05215.
13. Francesca Cibrario, Or Samimi Golan, Giacomo Ranieri, Emanuele Dri, Mattia Ippoliti, Ron Cohen, Christian Mattia, Bartolomeo Montrucchio, Amir Naveh, and Davide Corbelletto, ”Quantum Amplitude Loading for Rainbow Options Pricing”, Accessed at https://arxiv.org/abs/2402.05574.

Appendix

1) State Preparation for 4 Price Paths

State	Distribution Expected	Distribution Obtained
0	0.2	0.2043
1	0.35	0.3503
2	0.25	0.2486
3	0.2	0.1968

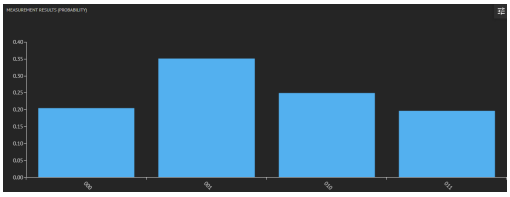


Figure 6: **Execution results for Option with 4 Price paths (without circuit width constraint)**

2) State Preparation for 8 Price Paths

State	Distribution Expected	Distribution Obtained
0	0.04	0.0394
1	0.2	0.2069
2	0.17	0.1705
3	0.25	0.2424
4	0.08	0.0781
5	0.05	0.0518
6	0.18	0.1813
7	0.03	0.0296

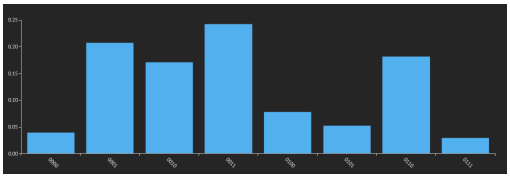


Figure 7: **Execution results for Option with 8 Price paths (without circuit width constraint)**