Quantum Credit Risk Analysis

Authors: Arjun Puppala, Jose Ramon Aleman, Marti Ciurana

Tutor: Giulio Gasbarri

Abstract

We introduce and examine quantum algorithms based on Quantum Amplitude Estimation (QAE), for estimating credit risk that outperforms Monte Carlo simulations on traditional computers. We calculate the Economic Capital (EC) required, which is the difference between the Value at Risk (VaR) and the predicted loss distribution value (LDV). The calculation of the economic capital required is of interest, because it quantifies the amount of capital required to be solvent at a certain level of confidence, making it an important risk statistic. We implement this problem for a realistic loss distribution and analyze its scaling to a realistic problem size. We present estimates of the overall number of necessary qubits, the expected circuit depth, and how this translates into an expected runtime on future fault-tolerant quantum hardware under acceptable assumptions.

1. Introduction

Credit risk analysis is a form of analysis performed by a credit analyst to determine a borrower's ability to meet their debt obligations. The purpose of credit analysis is to determine the creditworthiness of borrowers by quantifying the risk of loss that the lender is exposed to. The three factors that lenders use to quantify credit risk include the probability of default, loss given default, and exposure at default. [1]

The use of quantum computing could allow faster and more accurate solutions, for example determining a credit risk profile of a client. It is said that financial institutions that can make the most of quantum computing are likely to see significant benefits. Banks and asset managers optimize portfolios based on computationally intense models that process large sets of variables. For this reason, they will be able to more effectively analyze large or unstructured data sets.

In valuation, for example, the ability to speedily identify an optimal risk-adjusted portfolio is likely to create significant competitive advantage. For loan and bond portfolios, more precise estimates of credit exposures should lead to better optimization decisions.

American and European banks are also exploring quantum computing opportunities. [2]

- Bank of America strategist said quantum computing would be "as revolutionary in the 2020s as smartphones were in the 2010s."

- BBVA has formed a partnership to explore portfolio optimization and more efficient Monte Carlo modelling.
- Caixa Bank is running a trial hybrid framework of quantum and conventional computing with the aim of better classifying credit risk profiles.

2. Description of the Problem

Default risk is the risk that a lender takes on in the chance that a borrower will be unable to make the required payments on their debt obligation.

Whenever a lender extends credit to a borrower, there is a chance that the loan amount will not be paid back. The measurement that looks at this probability is the default risk. Default risk does not only apply to individuals who borrow money, but also to companies that issue bonds and due to financial constraints, are not able to make interest payments on those bonds. Whenever a lender extends credit, calculating the default risk of a borrower is crucial as part of its risk management strategy. Whenever an investor is evaluating an investment, determining the financial health of a company or private person is crucial in gauging investment risk.

Lenders generally examine a company's financial statements and employ several financial ratios to determine the likelihood of debt repayment. Free cash flow is the cash that is generated after the company reinvests in itself and is calculated by subtracting capital expenditures from operating cash flow. Free cash flow is used for things such as debt and dividend payments. A free cash flow figure that is near zero or negative indicates that the company may be having trouble generating the cash necessary to deliver on promised payments. This could indicate a higher default risk.

The risk of default on a debt arises from a borrower failing to make the required payments, for example:

- A consumer may fail to make a payment due on a mortgage loan, credit card, line of credit, or other loan.
- A company is unable to repay asset-secured fixed or floating charge debt.
- A business or consumer does not pay a trade invoice when due

2.1. Problem Definition

We analyze the credit risk of a portfolio of K assets. The default probability of every asset k follows a Gaussian Conditional Independence model, i.e., given a value z sampled from a latent random variable Z following a standard normal distribution, the default probability of asset k is given by:

$$p_k(z) = F\left(\frac{F^{-1}(p_k^0) - \sqrt{\rho_k}z}{\sqrt{1-\rho_k}}\right)$$

where F denotes the cumulative distribution function of Z, p_k^0 is the default probability of asset k for $z = 0$ and ρ_k is the sensitivity of the default probability of asset k with respect to Z. Thus, given a concrete realization of Z the individual default events are assumed to be independent from each other.

We are interested in analyzing risk measures of the total loss

$$L = \sum_{k=1}^{K} \lambda_k X_k(Z)$$

where λ_k denotes the loss given default of asset k, and given Z, $X_k(Z)$ denotes a Bernoulli variable representing the default event of asset k. More precisely, we are interested in the expected value $\mathbb{E}[L]$, the Value at Risk (VaR) of L and the Conditional Value at Risk of L (also called Expected Shortfall).

Where VaR and CVaR are defined as

$$\mathrm{VaR}_\alpha(L) = \inf\{x \mid \mathbb{P}[L <= x] \geq 1 - \alpha\}$$

with confidence level $\alpha \in [0, 1]$, and

$$\mathrm{CVaR}_\alpha(L) = \mathbb{E}[L \mid L \geq \mathrm{VaR}_\alpha(L)]$$

2.2. Problem Parameters

Quantum Amplitude Estimation (QAE)1 is a fundamental quantum algorithm with the potential to achieve a quadratic speedup for many applications that are classically solved through Monte Carlo (MC) simulation. It has been shown that we can leverage QAE in the financial service sector, e.g., for risk analysis2,3 or option pricing4,5,6, and also for generic tasks such as numerical integration7. While the estimation error bound of classical MC simulation scales as $\mathcal{O}(1/\sqrt{M})$, where M denotes the number of (classical) samples, QAE achieves a scaling of $\mathcal{O}(1/M)$ for M (quantum) samples, indicating the aforementioned quadratic speedup.

The canonical version of QAE is a combination of Quantum Phase Estimation (QPE)8 and Grover's Algorithm. Since other QPE-based algorithms are believed to achieve exponential speedup, most prominently Shor's Algorithm for factoring9, it has been speculated as to whether QAE can be simplified such that it uses only Grover iterations without a QPE-dependency. Removing the QPE-dependency would help to reduce the resource requirements of QAE in terms of qubits and circuit depth and lower the bar for practical applications of QAE.

Recently, several approaches have been proposed in this direction. In ref. 10 the authors show how to replace QPE with a set of Grover iterations combined with a Maximum Likelihood Estimation (MLE), in the following called Maximum Likelihood Amplitude Estimation (MLAE). In ref. 11, QPE is replaced by the Hadamard test, analog to Kitaev's Iterative QPE12,13 and similar approaches14,15.

Both in refs. 10 and 11 propose potential simplifications of QAE but do not provide rigorous proofs of the correctness of the proposed algorithms. In ref. 11, it is not even clear how to control the accuracy of the algorithm other than possibly increasing the number of measurements of the evolving quantum circuits. Thus, the potential quantum advantage is difficult to compare and we will not discuss it in the remainder of this paper.

In ref. 16, another variant of QAE was proposed. There, for the first time, it was rigorously proven that QAE without QPE can achieve a quadratic speedup over classical MC simulation. Following16, we call this algorithm QAE, Simplified (QAES). Although this algorithm achieves the desired asymptotic complexity exactly (i.e., without logarithmic factors), the involved constants are very large, and likely to render this algorithm impractical unless further optimized—as shown later in this paper.

In the following, we propose a new version of QAE—called Iterative QAE (IQAE)—that achieves better results than all other tested algorithms. It provably has the desired asymptotic behavior up to a multiplicative $\log(2/\alpha \log 2(\pi/4\epsilon))$ factor, where ϵ > 0 denotes the target accuracy, and 1 – α the resulting confidence level.

Like in ref. 16, our algorithm requires iterative queries to the quantum computer to achieve the quadratic speedup and cannot be parallelized. Only MLAE allows the parallel execution of the different queries as the estimate is derived via classical MLE applied to the results of all queries. Although parallelization is a nice feature, the potential speedup is limited. Assuming the length of the queries is doubled in each iteration (like for canonical QAE and MLAE) the speedup is at most a factor of two since the computationally most expensive query dominates all the others.

With MLAE, QAES, and IQAE we have three promising variants of QAE that do not require QPE and it is of general interest to empirically compare their performance. Of similar interest is the question of whether the canonical QAE with QPE—while being (quantum) computationally more expensive—might lead to some performance benefits. To be able to better compare the performance of canonical QAE with MLAE, QAES, and IQAE, we extend QAE by a classical MLE postprocessing based on the observed results. This improves the results without additional queries to the quantum computer and allows us to derive proper confidence intervals.

QAE was first introduced in ref. 1 and assumes the problem of interest is given by an operator $\mathcal{A}$ acting on n + 1 qubits such that

The problem is defined by the following parameters:

1. Number of qubits used to represent Z, denoted by n_z
2. Truncation value for Z, denoted by $z_{\max}$, i.e., Z is assumed to take 2^{n_z} equidistant values in $\{-z_{\max}, \ldots, +z_{\max}\}$
3. Base default probabilities for each asset $p_0^k \in (0,1)$, $k = 1, \ldots, K$
4. Sensitivities of the default probabilities with respect to Z, denoted by $\rho_k \in [0,1)$
5. Loss given default for asset k, denoted by λ_k

6. Confidence level for VaR / CVaR $\alpha \in [0, 1]$.

3. Methods

3.1. How to Estimate the Risk

Value at risk (VaR) is a statistic that quantifies the extent of possible financial losses within a firm, portfolio, or position over a specific time frame.

A financial firm, for example, may determine an asset has a 3% one-month VaR of 2%, representing a 3% chance of the asset declining in value by 2% during the one-month time frame. The conversion of the 3% chance of occurrence to a daily ratio places the odds of a 2% loss at one day per month.

The CVar quantifies the average expected loss.

3.2. Classical Method: Monte Carlo

The Monte Carlo method is a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. The underlying concept is to use randomness to solve problems that might be deterministic in principle. Monte Carlo Simulation predicts a set of outcomes based on an estimated range of values versus a set of fixed input values.

In other words, a Monte Carlo Simulation builds a model of possible results by leveraging a probability distribution, such as a uniform or normal distribution, for any variable that has inherent uncertainty. It, then, recalculates the results over and over, each time using a different set of random numbers between the minimum and maximum values. In a typical Monte Carlo experiment, this exercise can be repeated thousands of times to produce a large number of likely outcomes.

Several combinations of different scenarios are tested and the result of each scenario is analyzed. Therefore it’s slow and expensive.

3.3. Grover's Algorithm

In this section, we introduce Grover's algorithm and how it can be used to solve unstructured search problems. This algorithm can speed up an unstructured search problem quadratically.

Suppose you are given a large list of N items. Among these items there is one item with a unique property that we wish to locate; we will call this one the winner w.

To find the the marked item using classical computation, one would have to check on average $\frac{N}{2}$ of these boxes, and in the worst case, all N of them. On a quantum computer, however, we can find the marked item in roughly $\sqrt{N}$ steps with Grover's algorithm.

Before looking at the list of items, we have no idea where the marked item is. Therefore, any guess of its location is as good as any other, which can be expressed in terms of a uniform superposition:

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle.$$

If at this point we were to measure in the standard basis $\{|x\rangle\}$, this superposition would collapse, according to the fifth quantum law, to any one of the basis states with the same probability of $\frac{1}{n} = \frac{1}{2^n}$

Our chances of guessing the right value w is therefore 1 in 2^n, as could be expected. Hence, on average we would need to try about $\frac{N}{2} = 2^{n-1}$ times to guess the correct item.

This algorithm has a nice geometrical interpretation in terms of two reflections, which generate a rotation in a two-dimensional plane. The only two special states we need to consider are the winner $|w\rangle$ and the uniform superposition $|s\rangle$. These two vectors span a two-dimensional plane in the vector space $\mathbb{C}^n$.

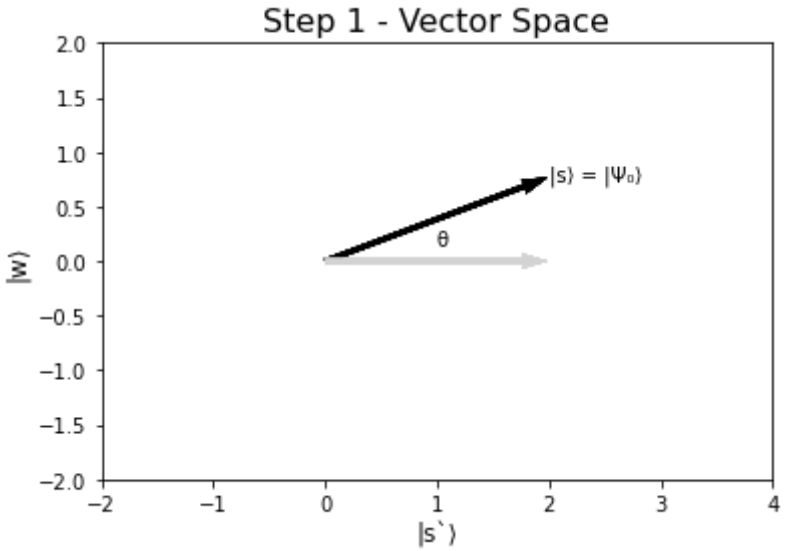
They are not quite perpendicular because $|w\rangle$ occurs in the superposition with amplitude $N^{-1/2}$ as well. We can, however, introduce an additional state $|s'\rangle$ that is in the span of these two vectors, which is perpendicular to $|w\rangle$ and is obtained from $|s\rangle$ by removing $|w\rangle$ and rescaling.

3.3.1. Algorithm Steps

3.3.1.1. Step 1 - Probabilities by Set

The amplitude amplification [6] procedure starts out in the uniform superposition $|s\rangle$, which is easily constructed from:

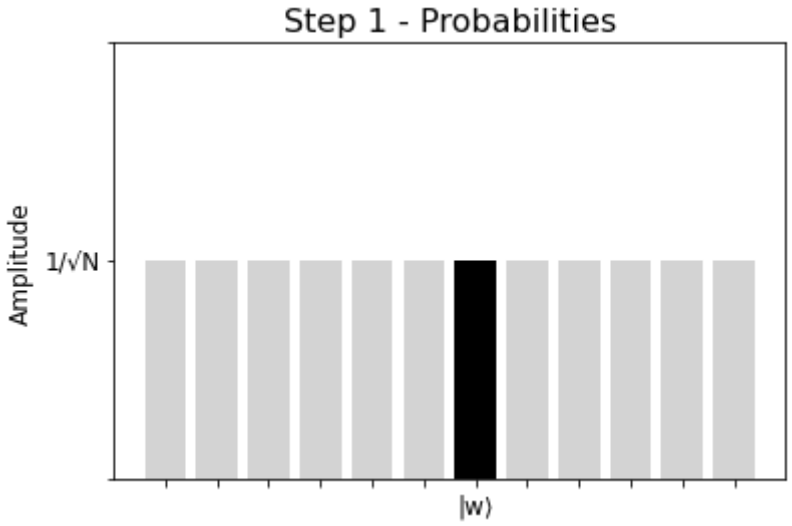
$$|s\rangle = H^{\otimes n}|0\rangle^n$$



> Corresponds to the two-dimensional plane spanned by perpendicular vectors $|w\rangle$ and $|s'\rangle$ which allows to express the initial state as:
>
> $$|s\rangle = \sin\theta|w\rangle + \cos\theta|s'\rangle$$

Where

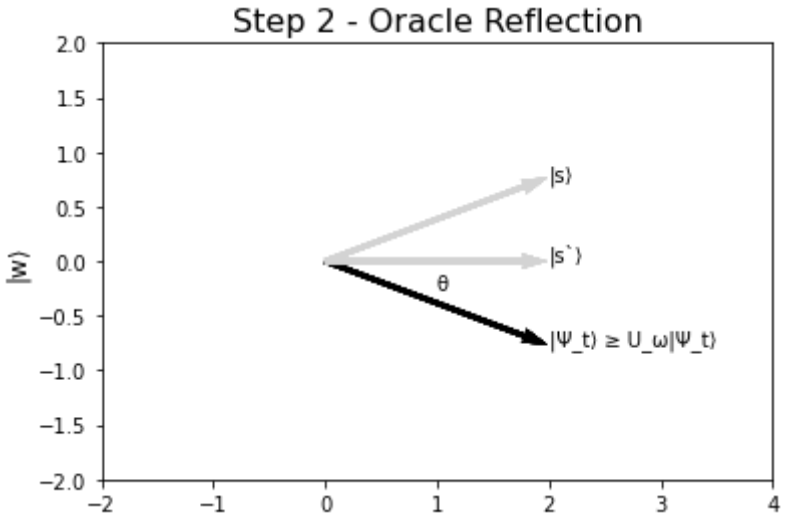
$$\theta = \arcsin\langle s|w\rangle = \arcsin\frac{1}{\sqrt{N}}$$



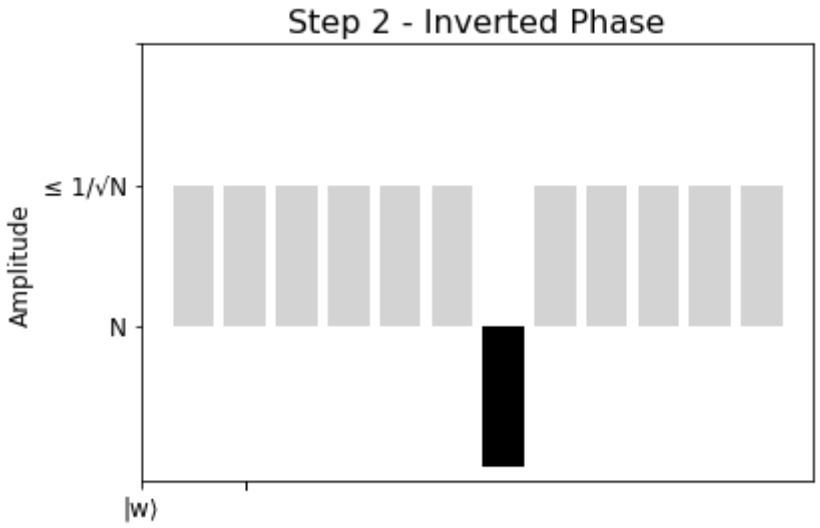
> Graph of the amplitudes of the state $|s\rangle$.

3.3.1.2. Step 2 - Oracle and Phase Inversion

We apply the oracle reflection U_f to the state $|s\rangle$:



> Geometrically this corresponds to a reflection of the state $|s\rangle$ about $|s'\rangle$.



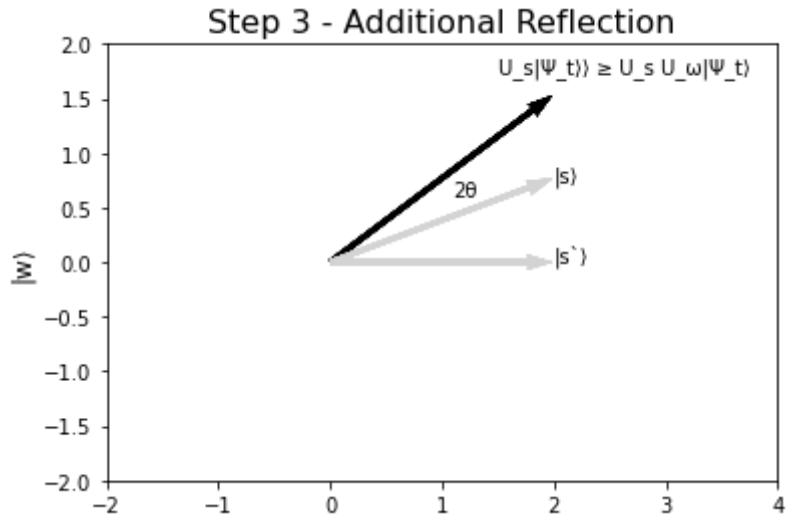
> This transformation means that the amplitude in front of the $|w\rangle$ state becomes negative, which in turn means that the average amplitude (indicated by a dashed line) has been lowered.

3.3.1.3. Step 3 - Reflection and Rotation

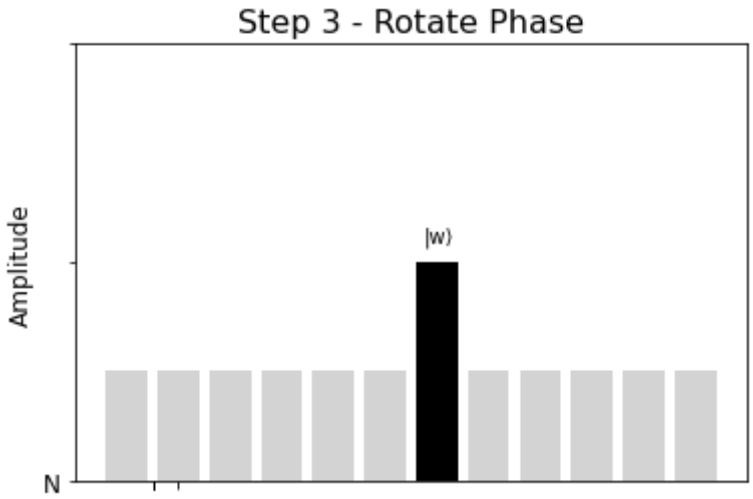
We now apply an additional reflection (U_s) about the state

$$|s\rangle : U_s = 2|s\rangle\langle s| - 1$$

This transformation maps the state to $U_s U_f |s\rangle$ and completes the transformation.



> Two reflections always correspond to a rotation.

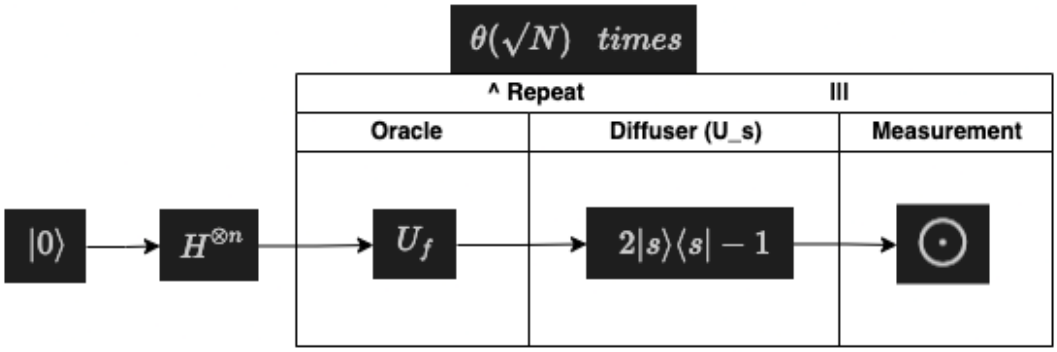


> The transformation $U_s U_f$ rotates the initial state $|s\rangle$ closer towards the winner $|w\rangle$

Two reflections always correspond to a rotation. The transformation $U_s U_f$ rotates the initial state $|s\rangle$ closer towards the winner $|w\rangle$. The action of the reflection U_s in the amplitude bar diagram can be understood as a reflection about the average amplitude. Since the average amplitude has been lowered by the first reflection, this transformation boosts the negative amplitude of $|w\rangle$ to roughly three times its original value, while it decreases the other amplitudes. We then go to step 2 to repeat the application. This procedure will be repeated several times to zero in on the winner.

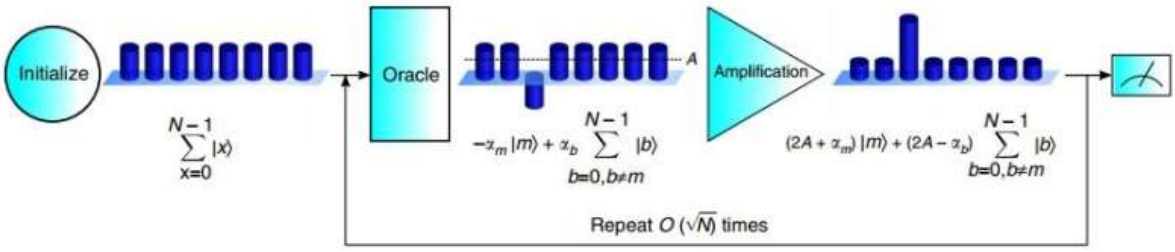
After t steps we will be in the state $|\psi t\rangle$ where:

$$|\psi_t\rangle = (U_s U_f)^t |s\rangle$$



3.3.2.1. Grover's Algorithm Search Implementation

The three stages of the 3-qubit Grover search algorithm: initialization, oracle, and amplification. [7]



3.4.4. Quantum Fourier Transform

The Fourier transform occurs in many different versions throughout classical computing, in areas ranging from signal processing to data compression to complexity theory. The quantum Fourier transform (QFT) is the quantum implementation of the discrete Fourier transform over the amplitudes of a wavefunction.

The discrete Fourier transform acts on a vector $(x0, \ldots, x_{N-1})$ and maps it to the vector $(y0, \ldots, y_{N-1})$ according to the formula:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk}$$

where: $\omega_N^{jk} = e^{2\pi i \frac{jk}{N}}$

Similarly, the quantum Fourier transform acts on a quantum state $|X\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$ and maps it to the quantum state $|Y\rangle = \sum_{k=0}^{N-1} y_k |k\rangle$ according to the formula:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk}$$

We see that only the amplitudes of the state were affected by this transformation.

This can also be expressed as the unitary matrix:

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega_N^{jk} |k\rangle\langle j|$$

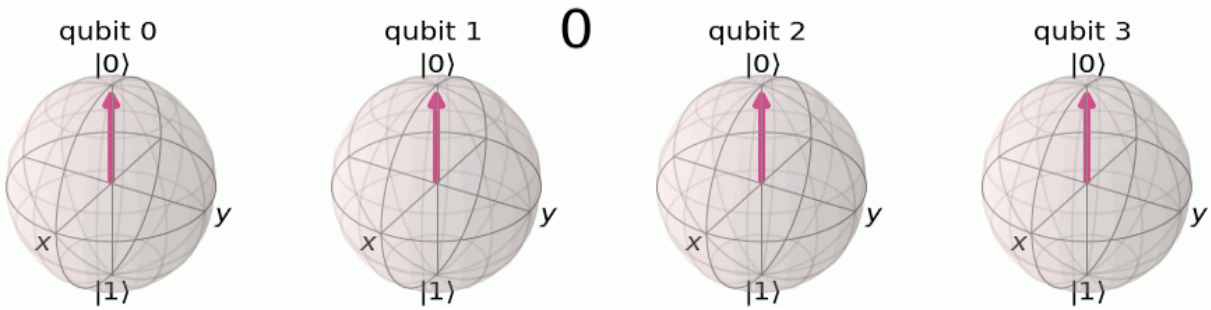
The quantum Fourier transform (QFT) transforms between two bases, the computational (Z) basis, and the Fourier basis. The H-gate is the single-qubit QFT, and it transforms between the Z-basis states $|0\rangle$ and $|1\rangle$ to the X-basis states $|+\rangle$ and $|-\rangle$. In the same way, all multi-qubit states in the computational basis have corresponding states in the Fourier basis. The QFT is simply the function that transforms between these bases.

$$|\text{State in Computational Basis}\rangle \xrightarrow{\text{QFT}} |\text{State in Fourier Basis}\rangle$$

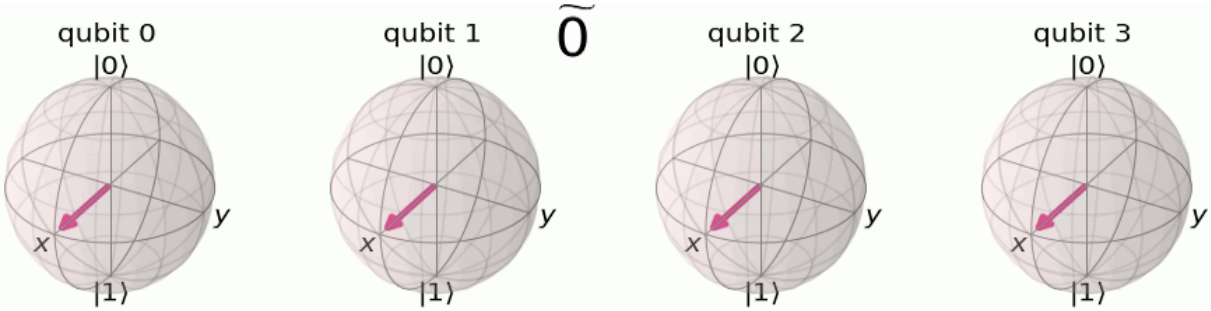
(We note states in the Fourier basis using the tilde (~)).

3.4.4.1. Counting in the Fourier basis:

In the computational basis, we store numbers in binary using the states $|0\rangle$ and $|1\rangle$:



The frequency with which the different qubits change; the leftmost qubit flips with every increment in the number, the next with every 4 increments, the third with every 4 increments, and so on. In the Fourier basis, we store numbers using different rotations around the Z-axis:



The number we want to store dictates the angle at which each qubit is rotated around the Z-axis. In the state $|\tilde{0}\rangle$, all qubits are in the state $|\widetilde{+}\rangle$. As seen in the example above, to encode the state $|\tilde{5}\rangle$ on 4 qubits, we rotated the leftmost qubit by $\frac{5}{2^n} = \frac{5}{16}$ full turns ($\frac{5}{16} \times 2\pi$ radians). The next qubit is turned double this radians, $\frac{10}{16} \times 2\pi$, this angle is then doubled for the qubit after, and so on.

Again, note the frequency with which each qubit changes. The leftmost qubit (qubit 0) in this case has the lowest frequency, and the rightmost the highest.

4. Results

We went on to show how QAE can be applied to the task of pricing an asset using real quantum hardware. We chose the simple model of a treasury bill whose value depends only on the interest rate. This simple experiment on 5 qubits confirmed the theoretical convergence rate of QAE and underlines its potential advantage compared to classical Monte Carlo methods.

In order to further demonstrate the capabilities of our algorithm, we used classical simulations of quantum hardware to show that it can also be applied to speed up the computation of risk measures of a simple two-asset portfolio. Nevertheless, we notice that to achieve quantum advantage in a real-world scenario, the quality of current quantum hardware needs to be improved. Errors arising from the limited coherence time and cross-talk when measuring the states of qubits need to be substantially suppressed. Furthermore, the number of qubits must be increased.

However, the current pace of advances in research directed at improving both quantum hardware and quantum algorithms makes us optimistic that real quantum advantage in risk analysis can be achieved in the future.

4.1. Uncertainty Model

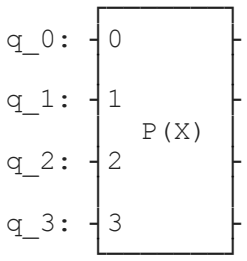
We now construct a circuit that loads the uncertainty model. This can be achieved by creating a quantum state in a register of n_z qubits that represents Z following a standard normal distribution. This state is then used to control single qubit Y-rotations on a second qubit register of K qubits, where a $|1\rangle$ state of qubit k represents the default event of asset k . [3]

The resulting quantum state can be written as

$$|\Psi\rangle = \sum_{i=0}^{2^{n_z}-1} \sqrt{p_z^i}|z_i\rangle \bigotimes_{k=1}^{K} \left(\sqrt{1-p_k(z_i)}|0\rangle + \sqrt{p_k(z_i)}|1\rangle\right)$$

where we denote by z_i the i-th value of the discretized and truncated Z [5].

Out[9]:

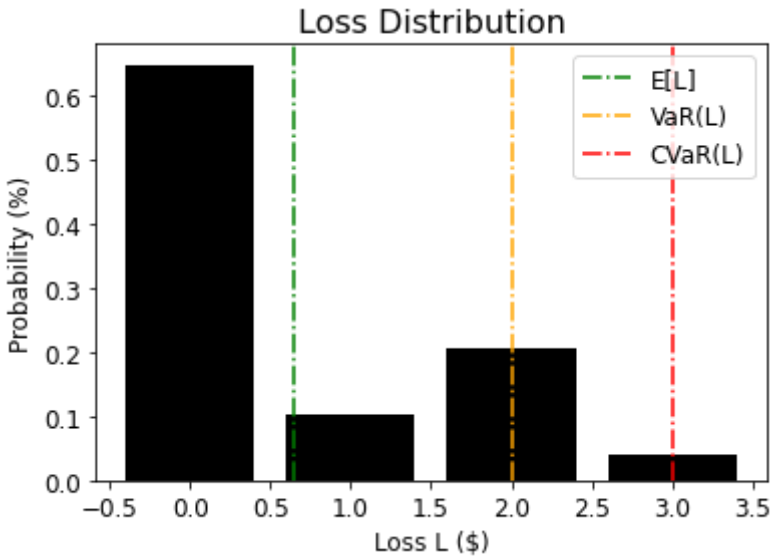


We now use the simulator to validate the circuit that constructs $|\Psi\rangle$ and compute the corresponding exact values for:

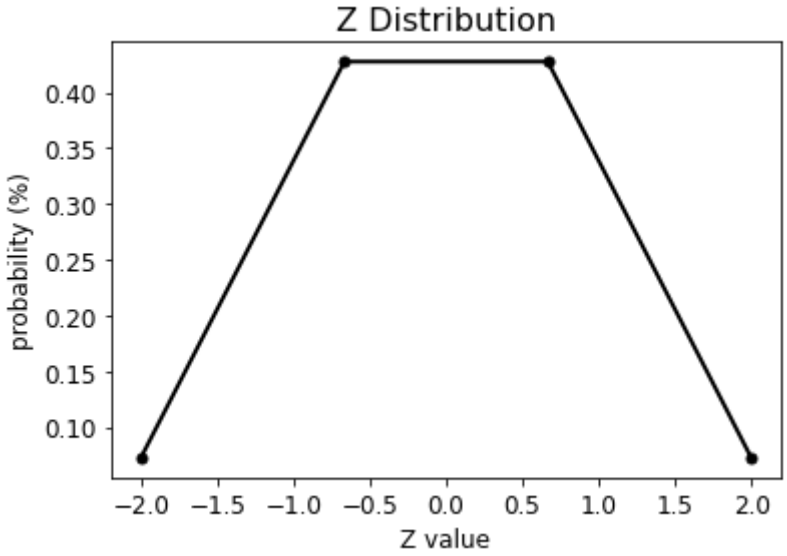
- Expected loss $\mathbb{E}(L)$
- PDF and CDF of L
- Value at Risk $VaR(L)$ and corresponding probability
- Conditional Value at Risk $CVaR(L)$

```
P[L] <= VaR[L]]:                  0.9591
Expected Loss E[L]:               0.6409
Value at Risk VaR[L]:             2.0000
Conditional Value at Risk CVaR[L]: 3.0000
```

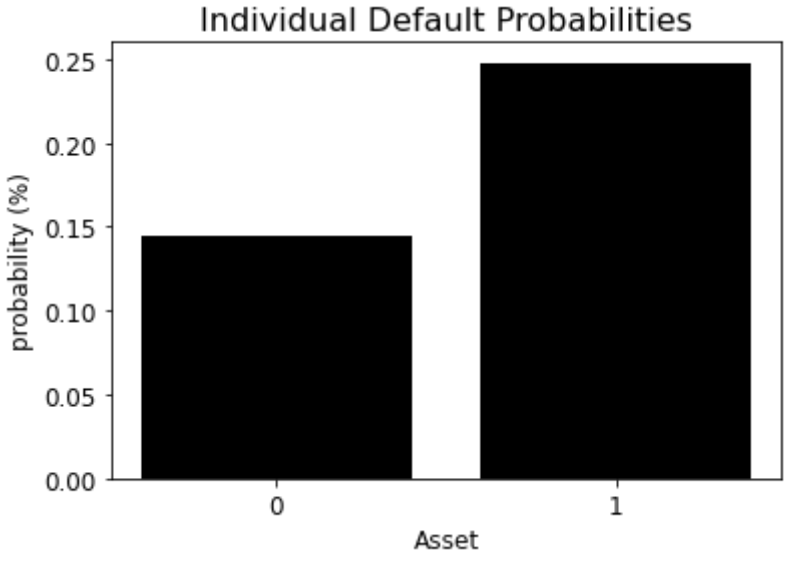
4.1.1. Loss Distribution



4.1.2. Z Distribution



4.1.3. Individual Default Probabilities



4.2. Expected Loss

To estimate the expected loss, we first apply a weighted sum operator to sum up individual losses to total loss:

$$\mathcal{S}: |x_1, \ldots, x_K\rangle_K |0\rangle_{n_S} \mapsto |x_1, \ldots, x_K\rangle_K |\lambda_1 x_1 + \ldots + \lambda_K x_K\rangle_{n_S}$$

The required number of qubits to represent the result is given by

$$n_s = \lfloor \log_2(\lambda_1 + \ldots + \lambda_K) \rfloor + 1$$

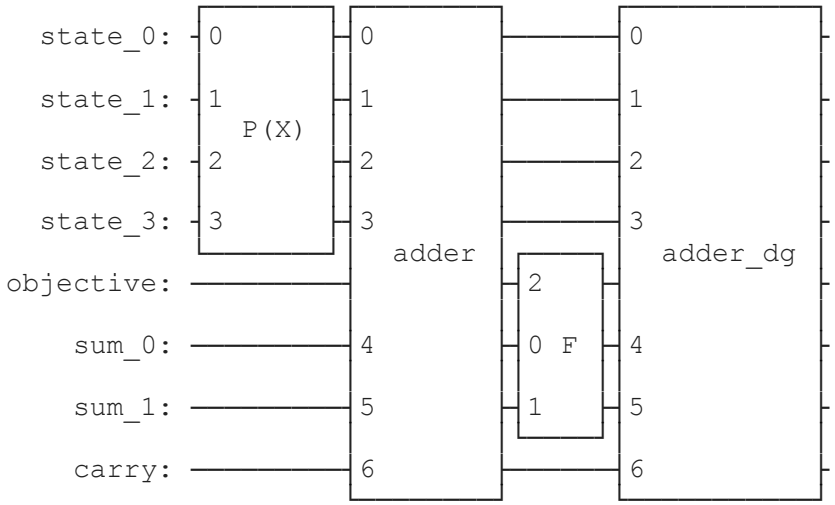
Once we have the total loss distribution in a quantum register, we can use the techniques described [4] to map a total loss $L \in \{0, \ldots, 2^{n_s} - 1\}$ to the amplitude of an objective qubit by an operator

$$|L\rangle_{n_s} |0\rangle \mapsto |L\rangle_{n_s} \left(\sqrt{1 - L/(2^{n_s} - 1)} |0\rangle + \sqrt{L/(2^{n_s} - 1)} |1\rangle \right)$$

which allows to run amplitude estimation to evaluate the expected loss.

Create the state preparation circuit:

Out[15]:



Before we use QAE to estimate the expected loss, we validate the quantum circuit representing the objective function by just simulating it directly and analyzing the probability of the objective qubit being in the $|1\rangle$ state, i.e., the value QAE will eventually approximate.

```
Exact Expected Loss:   0.6409
Exact Operator Value:  0.3906
Mapped Operator value: 0.6639
```

Next we run QAE to estimate the expected loss with a quadratic speed-up over classical Monte Carlo simulation.

```
Exact value:            0.6409
Estimated value:        0.6793
Confidence interval:    [0.6241, 0.7345]
```

4.3. Cumulative Distribution Function

Instead of the expected loss (which could also be estimated efficiently using classical techniques) we now estimate the cumulative distribution function (CDF) of the loss. Classically, this either involves evaluating all the possible combinations of defaulting assets, or many classical samples in a Monte Carlo simulation. Algorithms based on QAE have the potential to significantly speed up this analysis in the future.

To estimate the CDF, i.e., the probability $\mathbb{P}[L \leq x]$, we again apply $\mathcal{S}$ to compute the total loss, and then apply a comparator that for a given value x acts as

$$\mathcal{C} : |L\rangle_n|0> \mapsto \begin{cases} |L\rangle_n|1> & \text{if} \quad L \leq x \\ |L\rangle_n|0> & \text{if} \quad L > x \end{cases}$$

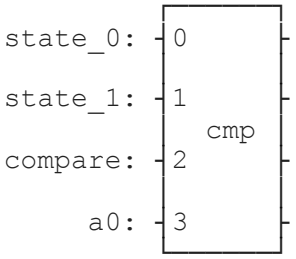
The resulting quantum state can be written as

$\sum_{L=0}^{x} \sqrt{p_L}|L\rangle_{n_s}|1\rangle + \sum_{L=x+1}^{2^{n_s}-1} \sqrt{p_L}|L\rangle_{n_s}|0\rangle$

Where we directly assume the summed up loss values and corresponding probabilities instead of presenting the details of the uncertainty model.

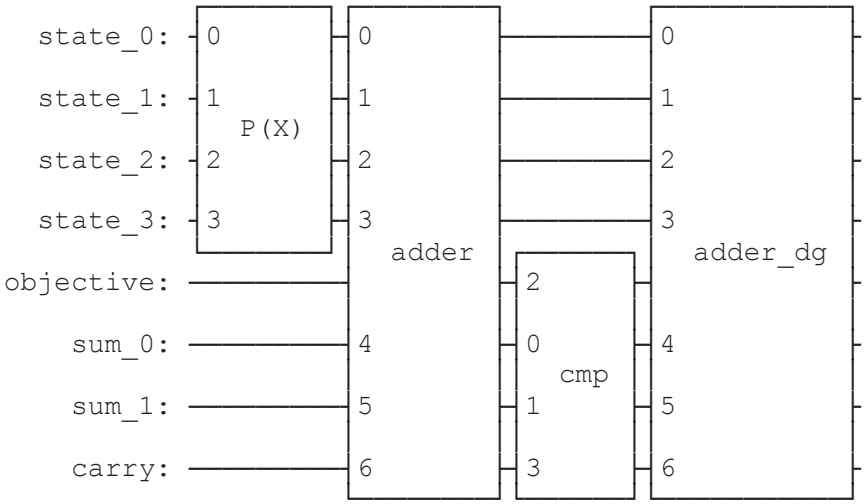
The CDF (x) equals the probability of measuring $|1\rangle$ in the objective qubit and QAE can be directly used to estimate it.

Out[18]:



Again, we first use quantum simulation to validate the quantum circuit.

Out[20]:




```
Operator CDF(2) = 0.9591
Exact    CDF(2) = 0.9591
```

Next we run QAE to estimate the CDF for a given x.

```
Exact value:            0.9591
Estimated value:        0.9599
Confidence interval:    [0.9588, 0.9610]
```

4.4. Value at Risk

In the following we use a bisection search and QAE to efficiently evaluate the CDF to estimate the value at risk.

```
.....................................................................
 Start bisection search for target value: 0.950
.....................................................................
low_level    low_value    level    value    high_level    high_value
.....................................................................
-1           0.000         1       0.753     3             1.000
 1           0.753         2       0.959     3             1.000
.....................................................................
 Finished bisection search
.....................................................................
```

```
Estimated Value at Risk:  2
Exact Value at Risk:      2
Estimated Probability:    0.959
Exact Probability:        0.959
```

4.5 Conditional Value at Risk

Last, we compute the CVaR, i.e. the expected value of the loss conditional to it being larger than or equal to the VaR. To do so, we evaluate a piecewise linear objective function $f(L)$, dependent on the total loss L, that is given by

$$f(L) = \begin{cases} 0 & \text{if} \quad L \leq VaR \\ L & \text{if} \quad L > VaR. \end{cases}$$

To normalize, we have to divide the resulting expected value by the VaR-probability, i.e. $\mathbb{P}[L \leq VaR]$.

Out[24]:

```
q479_0: ┤0    ├
q479_1: ┤1    ├
  q480: ┤2 F  ├
  a4_0: ┤3    ├
  a4_1: ┤4    ├
```

Out[25]:

```
<qiskit.circuit.instructionset.InstructionSet at 0x10afc3500>
```

Again, we first use quantum simulation to validate the quantum circuit.

```
Estimated CVaR: 3.2766
Exact CVaR:     3.0000
```

Next we run QAE to estimate the CVaR.

```
Exact CVaR:     3.0000
Estimated CVaR: 3.2959
```

5. Conclusions

By now, most people have heard that quantum computing is a revolutionary technology that leverages the bizarre characteristics of quantum mechanics to solve certain problems faster than regular computers can. Those problems range from the worlds of mathematics to retail business, and physics to finance.

But, as we have seen this year, Quantum computers are exceedingly difficult to engineer, build and program. As a result, they are crippled by errors in the form of noise, faults and loss of quantum coherence.

While classical computers are also affected by various sources of errors, these errors can be corrected with a modest amount of extra storage and logic. Quantum error correction schemes do exist but consume such a large number of qubits (quantum bits) that relatively few qubits remain for actual computation. So when only a couple of qubits are required it could be feasible, but when the number required increases the computing is unviable. That reduces the size of the computing task to a tiny fraction of what could run on defect-free hardware.

So, all in all, the application of this quantum algorithm (or quantum computing in general) is not feasible due to the noise and the quantity of qubits required.

6. References

- [1] Purpose of Credit Risk Analysis (corporatefinanceinstitute.com)
- [2] How Quantum Computing Could Change Financial Services (mckinsey.com)
- [3] Credit Risk Analysis (qiskit)
- [4] Quantum Risk Analysis. Stefan Woerner, Daniel J. Egger. (2019) (nature)
- [5] Credit Risk Analysis using Quantum Computers. Egger et al. (2019) (arxiv.org)
- [6] Quantum Amplitude Amplification and Estimation. Gilles Brassard et al. (2000) (arxiv.org)
- [7] Three stages of the 3-qubit Grover search algorithm (phys.org)
- [8] Iterative quantum amplitude estimation (nature)