Analysis of the Quantum Advantages for Deep Hedging

Soham Deshpande
Supervisor: Dr. Srinandan Dasmahapatra
Second Supervisor: Dr. Hansung Kim

April 2025

A project report submitted for the award of MEng Computer Science

Abstract

Parameterised Quantum Circuits (PQCs) have opened many doors, one such being the use in financial markets. In this paper, I look at the problem of developing an accurate market generator through the use of quantum computing for the purposes of hedging. Given a Quantum Circuit Born Machine (QCBM), we are able to exploit the high expressibility to generate synthetic data that mimics the statistical distribution of the original dataset. The market generator can then used to simulate an underlying asset to maturity with the intent of learning an optimal hedging strategy, a showcase of a data-driven approach to hedging exposure. I show that the synthetic data produced by this method has shown to capture the fat tails of the market better than classical methods, as well as demonstrating superiority in out-of-sample testing with COVID data. Different generator architectures have been compared to maximise the quality of the synthetic data and avoid issues with barren plateaus. The findings of this research will contribute to the growing literature on risk management in quantitative finance, with applications of the market generator extending beyond deep hedging.

Contents

1 Problem Statement

The problem of hedging a portfolio of derivatives is an important part of risk management used widely in financial institutions. This involves understanding the exposure to the market, and taking strategic positions to negate some of this risk. In an ideal world we can picture a perfect, frictionless market where transaction costs are negligible and every asset in the space has a price; here we can price and hedge perfectly. Unfortunately in practice, we experience incomplete markets due to frictions, costs that interfere with trades such as transaction costs or imperfect information. In addition, recent years have presented markets with periods of heightened volatility, much that disobey traditional frameworks. This generates the need for complex, realistic market models that can account for these.

Traditional methods of hedging options has shown to be ineffective for equity markets, new information resulting in rapid changes. Much of the available literature models the market as a smooth, continuous stochastic process within a Gaussian space. Such models are sufficient for common market activity but fail when presented with discontinuous moves in price. These can be reactions to geopolitical events or natural disasters; traditional models are incapable of capturing the effects. The introduction of Jump-diffusion models aimed to solve this issue though face similar issues. In reaction, we have recently observed non-parametric models that harness neural networks and machine learning which aim to demonstrate high accuracy on out-of-sample forecasts.

An alternative approach that has recently emerged utilises the power of quantum computing. The introduction of parameterised quantum circuits(PQCs) have opened up new pathways for navigating complex, large scale time series datasets. Unlike classical models that assume independence between observations, quantum models inherently operate within Hilbert space, where entangled bits naturally encode correlations and non-local dependencies, therefore offer a rich representational capacity.

At the heart of this capability is the Born rule, a foundational principle that governs how measurement probabilities are derived from the amplitudes of a quantum state. This non-linear mapping from complex amplitudes to real probabilities allows the model to intrinsically learn probabilistic distributions rather than pointwise outputs.

Furthermore, quantum entanglement allows multiple qubits to represent and propagate join dependencies that would require exponential resources to simulate classically. Within financial time series, where shocks and jumps can affect multiple assets non-linearly, quantum circuits can more naturally encode this behaviour. This naturally makes these models well-suited to model high volatility, high peaks, and Black-Swan events that are poorly modelled by continuous Gaussian processes.

In this research, I aim to tackle the problem of generating synthetic financial data, addressing issues that come about from using a classical method, particularly the estimation of tail risk and skewness. Through comparisons between traditional approaches, I aim to demonstrate an advantage in the expressibility of Quantum Circuit Born Machines

(QCBMs); these will be described quantitatively using measures such as Value at Risk (VaR) and Conditional Value at Risk (CVaR). By performing out-of-sample tests on COVID and Oil stock price data, I aim to highlight the weaknesses of traditional models and showcase a quantum superiority. There will also be an exploration into the variety of architectures available for the QCBM, evaluating different ansatz designs. Where unusual behaviours due to the quantum nature occur, such as barren plateau, I will explore in greater detail as well as any circuit optimisation techniques that may present themselves as possible solutions.

This paper will aim to add to the existing literature on risk management for financial firms as well as providing a framework for generating synthetic data. In addition to that, I aim to extend to the QCBM research that exists currently, noting down any behaviours that may be of interest to the curious and potential experts in the field.

2 Related Literature

To place this research within the context of existing literature, we can split the project into 2 components: the market generator, and parameterised quantum circuits.

The work around deep hedging has evolved, moving away from Greek-based hedging towards a sturdier framework using machine learning. Here a lot of work is being done, with many papers emphasising on using neural networks for optimising delta and gamma exposure [1, 33]. Buehler introduced an approach, modelling trading decisions as neural networks instead of relying on parameterised models [8]. Subsequent advancements focussed on developing realistic market simulators. Wissel proposed a market model for path generation of options but this still employed risk-neutral diffusion[38]. Wiese then introduced a new dimension by using GANs to convert options into local volatility models with simpler no-arbitrage constraints. This focussed on the local stochastic nature of options [10, 45, 46]. Some approaches suggest using actor-critic reinforcement learning algorithms to solve for an optimal value function, a move towards searching for a global maximum over local risk management [7, 27].

Recent research explores using quantum computing to hedge portfolios, here the authors presented a quantum reinforcement learning method based on policy-search and distributional actor-critic algorithms. They proposed using a Quantum Neural Network to approximate the value of a given value function by predicting the expected utility of returns using compound and orthogonal layers which were built using Hamming-weight unitaries [23].

TO CHANGE: This helped overcome the barren plateau by ensuring the gradient variance does not vanish exponentially with qubit count.

Another method models the entire return distribution, leveraging parameterised circuits to learn categorical distributions and capture variability and tail risk [9, 12].

There is an immense amount of research being done on exploiting the benefits of quantum computing, recent advancements being in quantum algorithms. These claim to provide exponential speed-up over classical methods, though in reality, we see great complexity in state preparation, requiring $\Theta(2^n/n)$ circuit depth with n qubits or $\Theta(n)$ circuit depth with $\Theta(2^n)$ ancillary qubits[48]. Here we see hybrid models such as Born machines and Quantum generative adversarial networks boasting high generalisation ability [18, 20, 22].

There has also been research in harnessing back action from quantum weak measurements to enhance the ability of quantum machine learning algorithms. In quantum reservoir computing, the ability to retain information from past inputs plays a key role in processing temporal series and producing future predictions [16, 17, 19, 28].

3 Markets and Derivatives

The market, though inherently can be thought of as a completely random process, where bids and asks are fulfilled, can be modelled as a stochastic process. The aim of this chapter is to serve as a brief introduction and set up notation for later chapters.

3.1 Brownian Motion

To represent this stochasticity, we must employ techniques introduced by Norbert Wiener, the Wiener process, more commonly referred to as standard Brownian Motion. This framework allows us to model continuous random walks of our stock price. Formally, a standard Wiener process, W_t, is a stochastic process where

1. $W_0 = 0$
2. The process W_t has stationary, independent increments
3. $\forall t \in Z$, the random variable W_t is normally distributed, $N(0, t)$
4. The paths of W_t are continuous ensuring no jumps in the path trajectory

These assumptions will help us understand the shortfalls of traditional techniques.

3.1.1 Itô Process

Itô processes are crucial for understanding the mathematical set up for modelling our assets. Itô calculus allows us to extend our understanding of deterministic calculus to the realm of stochasticity.

First considering the Itô integral, this will allow us to integrate wrt. to Brownian Motion, a non-differential stochastic process. Let W_t be a standard Brownian Motion and $f(\omega, t)$ be a stochastic process adapted to the filtration, the value of $f(\omega, t)$ only depends on information available up to time t, generated by W_t. The Itô integral of f wrt W_t over the time horizon [0,T] becomes:

$$\int_0^T f(t) dW_t \tag{3.1}$$

Suppose X_t is an Itô process which can be defined as a stochastic process which is written in the form

$$X_t = X_0 + \int_0^t \mu(X_t, t) ds + \int_0^t \sigma(X_t, t) dW_t \tag{3.2}$$

This structure is the solution to an SDE in the form

$$dX_t = \mu(X_t, t) dt + \sigma(X_t, t) dW_t \tag{3.3}$$

where $\mu(X_t, t)$ is the drift term and $\sigma(X_t, t)$ is the diffusion term. We must also discuss Itô's lemma which will allow us to obtain closed form solutions in the next sections.

Consider X in the form 3.3. Let $f(X_t, t)$ be a twice continuously differentiable function in t and x. Then, the differential $df(X_t, t)$ is given by Itô's Lemma:

$$df(X_t, t) = \left(\frac{\partial f}{\partial t} + \mu(X_t, t)\frac{\partial f}{\partial x} + \frac{1}{2}\sigma^2(X_t, t)\frac{\partial^2 f}{\partial x^2} \right) dt + \sigma(X_t, t)\frac{\partial f}{\partial x} dW_t$$

(3.4)

3.1.2 Geometric Brownian Motion

We can extend Brownian Motion to Geometric Brownian Motion by exponentiating the BM; this is done to satisfy the condition that stock prices are non-negative. GBM is a specific type of Itô process as can be observed when modelling it. We can now consider a continuous time process $S(t)$ which satisfies the SDE

$$dS_t = \mu S(t)dt + \sigma S(t)dW_t \tag{3.5}$$

where μ is the drift parameter, σ is the volatility parameter, and W_t is a Wiener process. Using Itô's lemma we obtain the solution

$$S_t = S_0 \exp\left[(\mu - \frac{1}{2}\sigma^2)t + \sigma W_t \right] \tag{3.6}$$

3.2 Market

It would be wise to define a market for a further understanding of assumptions made by traditional models. Consider a market, this can be thought of as an adapted (n+1) dimensional Itô process $X(t) = (X_0(t), X_1(t), ..., X_n(t))$ where

$$dX_i = \mu_i(t, \omega)dt + \sigma_i(t, \omega)dB(t); \quad X_i(0) = x_i \tag{3.7}$$

and $X_i(t)$ is the price of asset i at a given time t. This is the set up for a classical market, used for hedging under a traditional model. We can also make the following assumptions about the market:

- The market is liquid, allowing the trade to execute instantaneously at a given price
- There is no bid-ask spread, the price to buy and sell is the same
- Trading actions taken have no impact on the price of the asset traded

3.3 Derivatives

A derivative refers to any financial instrument whose value is derived from an underlying security, the most fundamental being futures and options. It is common practice to refer to the given underlying security as just 'underlying'.

3.3.1 Futures

A futures contract is a contract that gives the right and obligation to buy a given asset i a specified time T at price K.

3.3.2 Options

The two types of options going to be explored are Puts and Calls; a Call option gives the owner the right but not the obligation to buy a given asset i at a specified price(Strike price) K at time T. Similar to the Call, a Put option gives the owner the right but not the obligation to sell a given asset i at a price K at time T. If the owner can exercise the option any time up to T, we call this an American option. For the purposes of this research, I will only be dealing with vanilla European options.
It is important to define the payoffs for both options:

$$C_T = max(0, S_T - K) \tag{3.8}$$

$$P_T = max(0, K - S_T) \tag{3.9}$$

where S_T is the price of the underlying at time T.

3.4 Market Data

In this research I will be focussing on hedging a portfolio consisting of a single asset, hence requiring a simulation of a single underlying.

3.4.1 Euro Stoxx 50

The Euro Stoxx 50 Index (SX5E) and relevant derivatives. This is a stock index of 50 stocks in the Eurozone. This index captures around 60% of the free-float market capitalisation of the Euro Stoxx Total Market Index which covers about 95% of the free-float market in the Eurozone[14]. Rationale behind choosing this index is the availability of data, options traded with SX5E as the underlying and the liquidity of the index.
Derivatives that are held in the portfolio to be hedged will include those that have SX5E as the underlying, examples are weekly, monthly, and quarterly expiration options. These are European-style so can only be exercised upon maturity. Data can be found on Bloomberg[5] and Refinitiv [26].

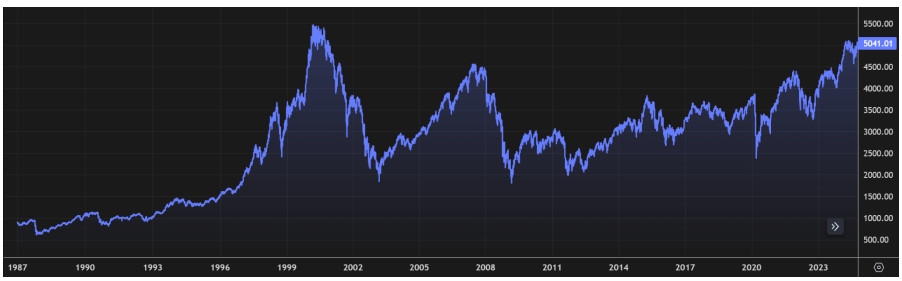

Figure 1: Euro Stoxx 50 price chart

3.4.2 Brent Crude Oil

Commodities often have complex dynamics, driven by geopolitical events and natural supply and demand. As well as this, oil in particular lends itself to increased volatility often responding to changes by the Federal Reserve and OPEC(Organisation of petroleum-exporting countries). These factors create characteristics such as heavy tails and jumps, often not being represented well by traditional models. I will be using Brent Oil as a benchmark asset to compare the expressibility of quantum derived models vs existing classical ones.

3.5 Hedging

The main issue being tackled in this report is the idea of optimal hedging. Hedging itself is the process of mitigating risk through trading other products. In this report, I am focussing on hedging an option with an underlying being the asset itself. Traditionally hedging involved deriving the exposure to the market, after buying an option, using a closed-form solution to classical models. The most commonly used is the delta exposure, derived from the Black-Scholes equation. The delta of an option is sensitive to the other parameters in BS, the one we're most concerned with being the underlying. Delta itself can be thought of as $\frac{\partial V}{\partial S}$, the change in the option price with respect to the underlying. The basic strategy for limiting risk is delta-hedging. This involves analysing the combined delta of the options bought and then buying the proportional amount in the underlying asset.

The biggest problems with delta-hedging include inaccurate assumptions about the underlying's behaviour, only hedging against the delta of the option, and the idea of when to hedge. One might take the strategy of hedging whenever the delta of the option reaches certain limits, called band-based hedging. This can be done efficiently if you are aware of how the option will react for its entire lifetime. Unfortunately we don't have such oracles that accurately predict the future, so we are left to mathematical assumptions and machine learning techniques to model such behaviour.

Deep hedging formulates hedging as a machine learning problem of determining the optimal points to hedge an option. Some papers treat this as a reinforcement problem, while others rely on time-series analysis. The common theme between most models available is the need for an accurate market generator. The underlying asset has to be simulated till maturity of the option to determine the best time to take a position. For this reason I will be focussing on the accuracy of the market generator in the context of deep hedging.

4 Merton-Jump Diffusion Model

My model of choice for comparison is the Merton-Jump Diffusion model, this an elementary model that goes beyond Black-Scholes by trying to capture the negative skewness and excess kurtosis of log price returns. This is done through the addition of a compound Poisson jump process. This aims to represent the jumps we observe in the market in a more realistic fashion rather than assuming constant volatility assumption made by Black-Scholes. As a simplification, I will be referring to Black-Scholes by BS and Merton-Jump Diffusion with MJD.

4.1 Black-Scholes

Let's start with the Black-Scholes model, an elementary model first proposed in 1973 by Robert Merton to price European vanilla options. [reference here]
Consider a call options on a non-dividend paying stock with expiry T and strike K. We will assume that the asset price will obey geometric brownian motion hence giving:

$$dS_t = \mu S_t dt + \sigma S_t dW_t \tag{4.1}$$

where W_t is a standard Brownian motion. We assume interest rates to be constant, meaning a unit of a given currency at time t will be worth e^{rt} at time t.
We can now consider the value of our call option $C(S_t, t)$, a twice differentiable function of stock price and time, by way of Itôs lemma we can say that

$$dC = \left(\frac{\partial C}{\partial t} + \mu S \frac{\partial C}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} \right) dt + \sigma S \frac{\partial C}{\partial S} dW \tag{4.2}$$

This is the SDE for the price of an option. In traditional hedging, we would use values derived from this SDE to construct a theoretical perfect hedge.

4.2 Model

A standard derivation of the model will allow us to explore its assumptions and limitations. This model consists of two components, jump and diffusion. The diffusion will be modelled using a Weiner process and log-normal jumps driven by a Poisson process. This gives us the following SDE.

$$dS_t = (\alpha - \lambda k) S_t dt + \sigma S_t dW_t + (y_t - 1) S_t dN_t \tag{4.3}$$

where W_t and N_t are Weiner and Poisson processes respectively. λ represents the intensity of the jumps, α is the drift rate (expected return), $y_t - 1$ is the relative jump size with k being the mean. Solving the equation gives us an exponential Lévy model described by

$$S_t = S_0 e^{\mathcal{L}_t} \tag{4.4}$$

where S_t is the stock price at time t, S_0 is the initial stock price. We can also define $\mathcal{L}_t$ to be

$$\mathcal{L}_t = (\alpha - \frac{\sigma^2}{2} - \lambda k)t + \sigma W_t + \sum_{i=1}^{N_t} Y_i \tag{4.5}$$

$\sum_{i=1}^{N_t} Y_i$ being the compound Poisson process.

4.3 Assumptions & Limitations

Through inspection of the equations, we can observe the following assumptions:

1. The asset price experiences continuous, random fluctuations over time, governed by Brownian motion (GBM)
2. The asset price experiences sudden, discontinuous jumps modelled by a Poisson process, occurring at a constant rate λ
3. Jumps sizes are assumed to be log-normal $ln(y_t) \sim \mathcal{N}(\mu, \sigma^2)$

Starting with the first assumption, we can see that assuming GBM may produce unrealistic behaviour, most important being a lack of excess kurtosis. Markets often exhibit fat tails, especially within commodities. One such event may be the release of news from OPEC+, the organisation of petroleum-exporting countries. A restriction in oil production may cause the price of oil to jump rapidly. In recent times, wars and conflict has also become ever present, causing large movements in asset prices; therefore it is not unrealistic to expect extreme price movements to be more frequent than can be modelled by a Gaussian.

The MJD requires calibration of parameters before use, typically done using historical data or implied volatility surfaces. Once calibrated these become assumptions of the data and so do not change even if the market observations move away from it. This would lead us to expect higher overfitting to the historical data, possibly failing in unseen conditions such as the market's reaction to COVID.

We also may expect poor volatility clustering with the MJD; constant volatility is not experienced by the market, instead periods of high volatility followed by periods of low volatility is observed. Though this paper won't be focussing on this phenomenon, it is important to consider.

4.4 Calibration

In this research, I have chosen to use maximum likelihood estimation to estimate the parameters for the MJD model. In the analytical solution we require five parameters: α, σ, μ_j, δ, and λ. These are the expected return, volatility of the given asset, expectation of the jump size, standard deviation of the jump size and lastly the jump intensity. We can then use MLE on the probability density of log returns $S_t = ln(\frac{S_t}{S_0})$

$$P(S_t) = \sum_{i=0}^{\infty} \frac{e^{-\lambda t}(\lambda t)^i}{i!} N(S_t; (\alpha - \frac{\sigma^2}{2} - \lambda k)t + i\mu_j, \sigma^2 t + i\delta^2) \tag{4.6}$$

The likelihood function hence becomes

$$L(\theta; S) = \prod_{t=1}^{T} P(S_t) \tag{4.7}$$

We can minimise the negative log-likelihood to obtain

$$-\ln L(\theta; S) = -\sum_{t=1}^{T} \ln P(S_t) \tag{4.8}$$

Another popular option to calibrate the MJD model is by considering the implied volatility surface of existing options. This technique can lead to a calibration but suffers with issues surrounding the sensitivity of the tails of the asset prices. It is also well documented that given a function that measures the calibration error, we can observe a largely flat landscape surrounding the optimal solution, implying obtaining accurate parameters can become very computationally expensive, often requiring hundreds of iterations [**jump05**]. These difficulties can translate into a poor hedge, leaving a buyer overexposed to market fluctuations.

5 Quantum Computing

This section aims to serve as a brief introduction to quantum computing.

5.1 Quantum Systems

Unlike classical computing, quantum computing acts in a non-deterministic manner, the computer remains in multiple states with given probabilities rather than a fixed resultant state as expected from classical computers. Formally we can define a qubit to be a quantum system where the states of 0 and 1 are represented by a pair of normalised and mutually orthogonal quantum states $|0\rangle$ and $|1\rangle$. Intuitively however, let's start with a two-state machine; we can describe such system to be in the state $|0\rangle$ with some amplitude α and in $|1\rangle$ with amplitude β. This can be represented as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{5.1}$$

for some α and β such that $|\alpha|^2 + |\beta|^2 = 1$; we can refer to this as a superposition of the states $|0\rangle$ and $|1\rangle$. If measured in the standard basis, we would expect the outcome to be $|k\rangle$ with a certain probability, this outcome resulting in the output state of the measurement gate to also be $|k\rangle$. This would mean our initial state $|\psi\rangle$ is irreversibly lost; we refer to this as a collapse of state.

Each qubit can be thought of as a vector, $\mathbf{v}$, on a Bloch's sphere which can be represented in two basis: θ and φ. θ is the angle between $\mathbf{v}$ and the z-axis. φ becomes the angle between $\mathbf{v}$ and the x-axis. Considering a more general parameterisation of 5.1 gives us

$$|\psi\rangle = \cos(\frac{\theta}{2})|0\rangle + \sin(\frac{\theta}{2})e^{i\varphi}|1\rangle \tag{5.2}$$

5.1.1 Tensor Products

Tensor products, $\otimes$, to allow us to combine vector spaces. Lets consider a two qubit system:

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle \text{ and } |\phi\rangle = \beta_0|1\rangle + \beta_1|1\rangle \tag{5.3}$$

their tensor product becomes:

$$|\psi\rangle \otimes |\phi\rangle = \alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle \tag{5.4}$$

5.1.2 N-qubit systems

The Bloch sphere, figure 2, allows us to visualise all the pure states and the rotations taken on $\mathbf{v}$. The space of all the possible orientations of ψ is called the Hilbert space. Formally a Hilbert space, $\mathcal{H}$, is a vector space which is an inner product space over $\mathcal{C}$ that is also complete with respect to the norm. For a single qubit the state space is $\mathbb{C}^2$,

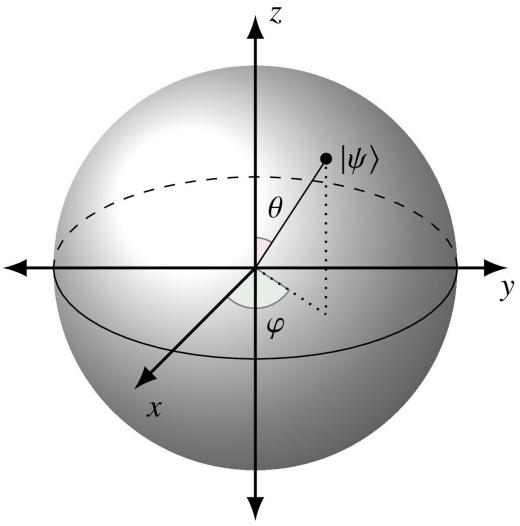

Figure 2: Bloch sphere

a 2-dimensional Hilbert space, satisfying 5.2. When extending to n qubits, the Hilbert space becomes

$$\mathcal{H}_n = \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes ... \otimes \mathbb{C}^2 = (\mathbb{C}^2)^{\otimes n} = \mathbb{C}^{2^n} \tag{5.5}$$

The computational basis for $\mathcal{H}_n$ consists of all the possible n-bit strings:

$$\{|x_1 x_2 ... x_n\rangle\}, \text{ where } x_i \in \{0, 1\} \tag{5.6}$$

There are 2^n such basis states, each representing a classical bit string (e.g. $|00...0\rangle$,$|00...1\rangle$,...,$|11...1|\rangle$).

5.1.3 Entanglement

Quantum entanglement is a fundamental feature of quantum mechanics, allowing qubits to be deeply correlated to the point their joint state cannot be represented as a product of their individual states. Measurement of one state will have an instantaneous impact on the other entangled qubit, determining the outcome for both qubits. Entanglement emerges from the structure of the tensor product. A state $|\psi\rangle \in \mathbb{C}^2 \otimes \mathbb{C}^2$ is entangled if there are correlations between measurement outcomes of each qubit that cannot be explained by any product of the independent states, we cannot decompose it into $|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$ for some $\psi_A \in \mathcal{H}_A$ and $|\psi_B\rangle \in \mathcal{H}_B$.

5.2 Quantum Operators

Before forming quantum circuits, we must first understand how quantum gates operate. In quantum computing, the evolution of quantum states is governed by unitary operations, linear transformations $U : \mathcal{H} \to \mathcal{H}$ satisfying $U^\dagger U = UU^\dagger = I$. These are implemented as quantum gates, acting on the qubits with single-qubit gates being represented by 2×2 unitary matrices, and multi-qubit by tensor products. There are many gates but the ones we will discuss the ones used in this project.

The Pauli group is a foundational set of operators that consists of three Pauli matrices and an identity. These form an orthonormal basis for 2×2 Hermitian matrices, $P = P^\dagger$ where $P^\dagger$ is the conjugate transpose, and are crucial for in defining measurements, quantum error correction, and stabiliser circuits. For a single qubit, we can define the Pauli group, $\mathcal{P}$, with phases $\{\pm 1, \pm i\}$ to be

$$\mathcal{P} = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\} \tag{5.7}$$

where

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{5.8}$$

We can extend this to the Pauli rotation gates which allow you to rotate a vector around the Bloch sphere. The first one, $R_x(\theta)$, allows us to rotate a qubit around the x-axis and is given by:

$$R_x(\theta) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$$

Likewise the R_y and R_z are defined similarly, allowing rotations around the y and z axes.

$$R_y(\theta) = \begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$

It is worth noticing the Pauli X gate is equivalent to a rotation of π in the x-axis, which can also be represented as $R_x(\pi)$. Another key gate is the CNOT gate, a two-qubit gate used to generate entanglement. The CNOT acts on a control and target qubit, flipping the target if the control qubit is in the $|1\rangle$ state.

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

These entangled states are crucial for learning complex distributions, allowing us to faithfully create synthetic market data.

5.3 Measurement

Measurement is a key part of quantum computing, allowing us to extract information from quantum systems through projections onto a particular basis. The basis that we measure in determines how we interpret the state of a given qubit. The most common one used is the computational basis(Z-basis), which consists of the two eigenstates of the Pauli-Z operator, states $|0\rangle$ and $|1\rangle$ for a single qubit system and tensor products of these single-qubit states when extending to multiple qubits. These states correspond to the eigenvalues 0 and 1, respectively. The Pauli-Z mathematically represents a measurement

along the z-axis of the Bloch sphere. When measuring, we are effectively projecting the quantum state onto these eigenvectors, with the measurement probabilities determined by squaring the magnitudes of state's components in the z-axis.

Measurement in the X-basis is similar, we are now projecting the quantum state onto the eigenstates of the Pauli-X operator. These are the $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ states.

5.4 Clifford and non-Clifford Gates

Often articles focus on the benefits of quantum computing being superposition and entanglement but we can think beyond that and consider those gates that can't be efficiently simulated classically, the untold heroes. The operations we have discussed previously such as Pauli X, $CNOT$ etc. can be put into two main categories: Clifford and non-Clifford. The Clifford group involves those gates that can be simulated efficiently classically, with non-Clifford containing those that are responsible for the quantum speed ups observed such as the T-gates.

Clifford group contains those gates that map Pauli operators to other Pauli operators under conjugation. We can formally say that for a Clifford unitary, U, and for every Pauli operator, P, $UPU^\dagger$ is still a Pauli operator. These, however, do not form a universal set of quantum gates so require non-Clifford operators. Though not able to represent every possible operation, they still serve a large purpose, typically being used in quantum error correction algorithms or for testing entanglement.

Non-Clifford gates are those that cannot be simulated efficiently on a classical machine, a common being the T-gate.

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} \tag{5.9}$$

An important property that defines this as non-Clifford is the fact that is doesn't preserve the Pauli group under conjugation. Formally

$$TXT^\dagger \neq \text{Pauli Operator} \tag{5.10}$$

Another important gate is the Toffoli gate which serves as a three-qubit gate that applies a NOT operation the third qubit if the first two qubits are in the $|1\rangle$ state.

Non-Clifford gates can be non-trivial to implement unfortunately. One such method for implementation of non-Cliffords includes magic state distillation, using ancilla bits to aid the computation, adding large computational costs. These also often require non-transversal methods which can increase the error propagation, making a fault tolerant device ever more difficult. For this reason, we would like a framework to reduce the T-gate count in a circuit.

5.5 Born Rule

An essential part of quantum computing involves the existence of the Born Rule. Given a quantum state $|\psi\rangle \in \mathcal{H}$, the Born rule states the probability of measuring an outcome $x \in \{0,1\}^n$ in the computational basis is

$$p(x) = |\langle x|\psi(\theta)\rangle|^2 \tag{5.11}$$

where

$$|\psi(\theta)\rangle = U(\theta)|0\rangle^{\otimes n} \tag{5.12}$$

This means that the probability distribution over all possible 2^n outcomes is derived from the squared magnitudes of the amplitudes in the quantum state vector. Unlike classical probabilistic models, where probabilities are assigned directly, quantum probabilities arise from the inner product between the quantum and measurement state, through the application of the Born rule. As such, the quantum devices are able to leverage superposition and interference effects that have no classical counterpart. This makes it suitable for representing probability distributions with non-trivial structures such as financial time-series with jumps, or fat-tailed behaviours.

LOOK INTO CHANGING:
The state $|\psi(\theta)\rangle$ is generated by evolving state $|0\rangle$ according to a Hamiltonian H that is constructed from gates. Once combined, the gates form a parameterised quantum circuit which is parameterised by using the variables governing each gate, θ. By tuning the values of θ_i one can allow for an evolution to any state that will serve as a solution to a given problem.
By taking the distribution associated to the state, $|\psi(\theta)\rangle$ we can treat the PQC as a generative model, upon measurement will generate samples of a target distribution χ. This model is parameterised by θ, which defines a quantum circuit $U(\theta)$ made up of a set of quantum gates. By measuring the circuit, we can obtain samples. Producing samples that emulate the target distribution involves minimising the parameters of the circuit $U(\theta)$, a process once convergence is reached, will generate accurate samples [25].

5.6 Parameterised Quantum Circuits

Parameterised quantum circuits(PQCs) are a powerful framework consisting of rotation and entanglement gates for hybrid quantum-classical algorithms. A PQC is a quantum circuit with rotation gates with parameters $\theta = (\theta_1, \theta_2 ... \theta_n)$ which can be tuned classically to give desired results. These rotations are often interleaved with entanglement gates such as CNOTs. The design of the PQC is dependant on the choice of ansatz used; an ansatz can be thought of as a trial state that is used as the starting point for the optimisations. It's an empty canvas with gates and parameters that need to be tuned to give meaning.

When thinking about PQCs, we must also discuss NISQ(Noisy Intermediate-Scale Quantum) devices. These are the current generation of quantum processors that have limited qubit counts, roughly 1000 qubits, but lack quantum error correction; this makes them

prone to noise and decoherence. This makes the devices very sensitive to external interference, especially from the environment. These computers aim to serve as a vehicle for research rather than providing a quantum advantage at this time.

The biggest challenge of NISQ devices is the high error rates which may render quantum algorithms useless due to the volume of noise. Noise can significantly worsen the barren plateau effect, expanded on further in this report, flattening the optimisation landscape. If the noise grows, it can have the effect of acting as random unitary operation, possibly causing gradients to vanish or making it difficult for classical optimisers to understand descent directions. Due to this, it is imperative to keep gate and entanglement counts to the acceptable minimum. This gives way to research on optimal circuits; I have tried to investigate the effect of the circuit ansatz on the end result, showing slight modifications to entanglement structures can have a large effect on the trainability and expressibility.

5.7 Basis Encoding

When modelled data is drawn from a continuous domain, log returns, the output space must be discretized to be representable in the computational basis. We achieve this through the process of basis encoding, involving binning the input data. Formally, given a continuous random variable, $X \in \mathbb{R}$, we discretize this into 2^n bins, where each bin corresponds to a basis state $|x\rangle$ for $x \in \{0,1\}^n$. This encoding induces a discrete probability distribution over the basis states which becomes the empirical target distribution for the training the quantum circuit; the PQC can then learn to approximate this histogram-like distribution using the Born rule. It is important to note that the resolution of the learnt distribution is therefore limited by the number of qubits, n, so should be a consideration when designing the PQC.

This process allows us to use classical statistical divergences such as Kullback-Leibler divergence for loss function since both the model and data reside in a common discrete sample space.

6 Quantum Circuit Born Machine

The Quantum Circuit Born Machine(QCBM) is an extension of PQCs, designed with the intent of probabilistic generation at its core. It consists of a PQC with trainable parameters, θ. These parameters can be tuned through an optimisation process such that sampling the PQC replicates a target distribution. This is the model that is leveraged to generate the synthetic market data.

Formally, given a dataset $D = \{x_1, x_2..x_n\}$ consisting of n market observations and obeys a given distribution χ_d, we would like the QCBM to replicate the distribution and, once sampled, generate synthetic data points that are of the distribution χ_s such that χ_s approximates χ_d.

Let $U(\theta)$ be a PQC acting on n qubits, where $\theta \in \mathcal{R}^m$ is a vector of tunable parameters. Initially, the circuit will prepare a quantum state:

$$|\psi(\theta)\rangle = U(\theta)|0\rangle^{\otimes n} \tag{6.1}$$

$|0\rangle^{\otimes n}$ representing all the qubits in the computational basis state $|0\rangle$.
Upon measurement in the computational basis, this state will produce a probability distribution according to Born's rule:

$$p_\theta(x) = |\langle x|\psi(\theta)\rangle|^2 \tag{6.2}$$

where $x \in \{0,1\}^n$ represents a bit string of measurement outcomes. The goal of the QCBM now becomes to find optimal parameters of θ such that the output probability distribution replicates that of the target, letting $p_\theta(x)$ approach χ_d.

6.1 Implementation

The distribution to be learnt could not simply be the price at each time stamp, as sampling the QCBM would remove any time component associated. Many ideas were tested but the final design involved learning the distribution of log returns. This was acceptable as we would be simulating many paths and less focussed on the given price at each point. Formally, we are learning r, the log returns, and propagating the series using

$$S_t = S_{t-1}e^{r_t} \tag{6.3}$$

At each time step, the QCBM is sampled to gain a log return to be used to generate the next value. Limitations of this can involve lack of volatility clustering which can make it unsuitable for time-series prediction, though uses a similar technique to classical methods. In the results section I explore whether this method does provide a better hedge or not.

Implementation of the QCBM involved a few more steps. The target distribution is of log returns, so the market data had to be converted into the appropriate format. Before using the QCBM, the input data must also be discretised. This involves binning the log returns into 2^n bins, where n is the number of qubits in the system. The value of n is

chosen based on the accuracy desired, with larger values of n allowing for more 'bins', leading to better expressibility. Making n too large can lead to issues with optimisation and giving way to the barren plateau phenomenon.

The training process then involves minimising a loss function, in this case using the Kullback-Leibler(KL) Divergence.

$$\mathcal{L}_{KL}(\theta) = \sum_{x \in \{0,1\}^n} q(x) log(\frac{q(x)}{p_{\theta(x)}}) \tag{6.4}$$

Optimisation of these parameters was left to the PennyLane[ref here] package. The optimiser chosen is Adam, a gradient-descent optimiser with adaptive learning rates.

The ansatz for this quantum circuit was varied, testing multiple architectures to explore the effects, expanded on further in section 6.2.

The final pipeline for the QCBM is as follows:

1. Pre-processing of market data
2. Binning/discretising the input data
3. Constructing the PQC for the QCBM
4. Comparison of QCBM's probability distribution to target distribution using KL Divergence
5. Optimisation of QCBM's parameters, θ
6. Analysis of synthetic distribution

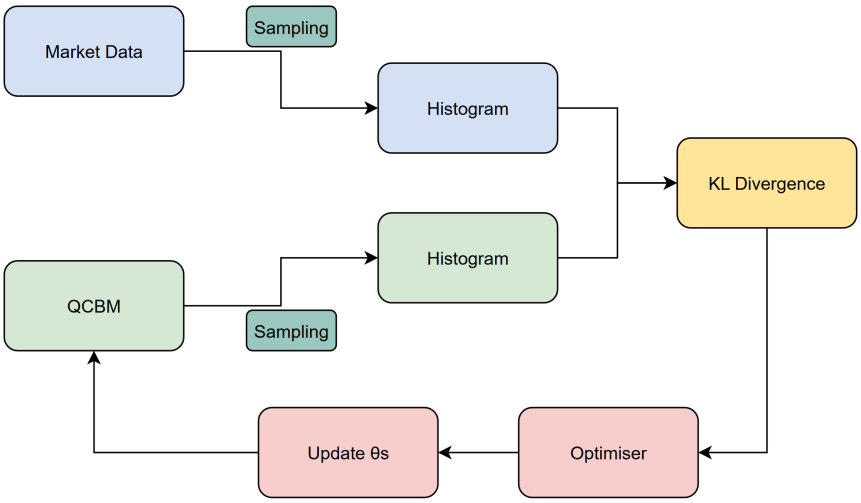

Figure 3: Overview of the QCBM design

6.2 Barren Plateau

A point of concern when searching for the optimal set of θs is the large search space, here we may observe issues such as barren plateau(BP). BP insists that the gradient of the parameters of a given PQC will vanish exponentially with respect to the search space. McClean[ref here] shows that for a PQC with L layers in the form:

$$U(\theta) = \Pi_L^{l=1} U_l(\theta_l) W_l \tag{6.5}$$

where $U_l(\theta_l)$ are parameterised gates and W_l are fixed gates, and with cost function:

$$E(\theta) = \langle 0| U(\theta)^\dagger H U(\theta) |0\rangle \tag{6.6}$$

where H is a Hermitian observable, the gradient variance vanishes exponentially, leaving us with the following relationship:

$$Var[\partial_k C(\theta)] \leq \mathcal{O}\big(\frac{1}{2n}\big) \tag{6.7}$$

for n-qubits.

This shows us to expect flat optimisation landscapes, leading to sub-optimal values of θ being learnt. This relationship also shows us why the ansatz chosen makes a big difference on the learnability, something I further explore in the next section.

6.3 Architectures

The design of the ansatz can significantly affect the ability to learn and represent the target distribution. The number of gates, depth, and entanglement structure all affect the expressibility and trainability of the circuit. Circuits with higher entanglement and parameterised gates are theoretically able to represent any distribution, though it comes at the cost of noise and barren plateaus. Choosing a simpler ansatz may converge quicker but results in a weaker approximation, oversimplifying the solution. The architecture will also affect the optimisation landscape; random parameterised circuits with deep, unstructured layers are more prone to barren plateaus, making training very difficult. Due to this we strive for a balance, one that is able to learn the complexities of the market without compromising on finding the optimal parameters.

6.3.1 Brick

The first architecture investigated is the Brick architecture. This ansatz arranges the gates in a staggered, layered pattern that resembles a brick wall. These have alternating layers of single-qubit rotation and nearest-neighbour hardware connectivity. This simple entanglement structure provided the best trainability, converging to a solution much faster than the other circuits. We can argue this was the case due to their local connectivity, on NISQ devices it is able to minimise cumulative errors. As well as this, the KL divergence loss was often much lower than the other circuits. For these reasons it was the circuit of choice for all the results in the following sections.

It must be said that research indicates this structure can face scaling limitations for problems that require long-range correlations. In the paper we are investigating correlations between 2 prices so are not affected by this property but it should be noted for any further developments made beyond this report. [ref: Empirical Comparisons on NISQ Devices].

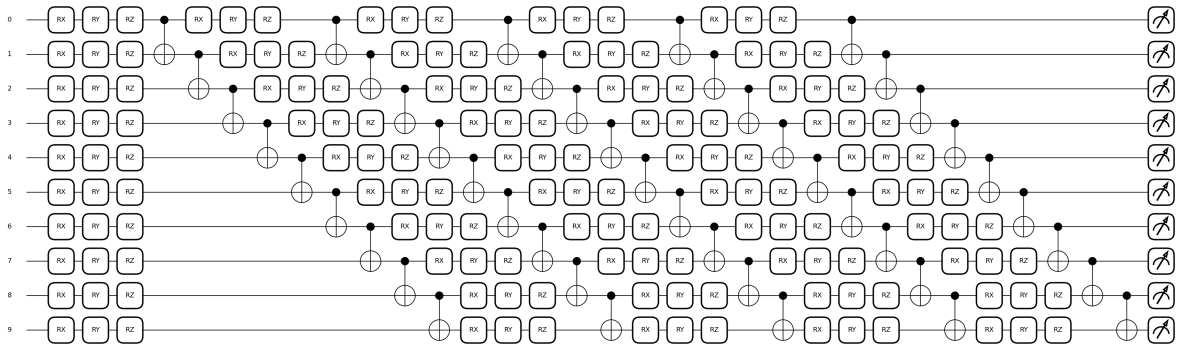

Figure 4: 10x5 Brick Ansatz

6.3.2 Pyramid

Though shown that the Brick ansatz provides the best results, it is important to test other circuits as well. The pyramid structure offers a simpler entanglement structure compared to the brick, also having nearest-neighbour hardware connectivity. This made trainability less of a concern but when looking at the KL divergence, it offered a worse loss indicating the easier optimisation came at the cost of expressibility. We could make conclusions that this circuit was not able to learn the appropriate correlations in the data, making it less effective than a traditional model.

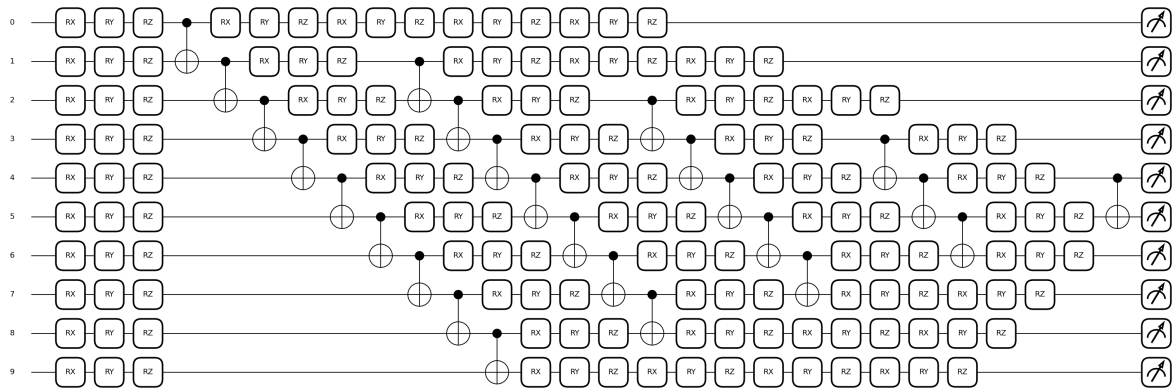

Figure 5: 10x5 Pyramid Ansatz

6.3.3 Butterfly

The Butterfly offers a more complex entanglement structure, with all-to-all hardware connectivity. This theoretically should provide less gate overhead but physical implementations experience practical issues. The physical wiring for all-to-all can prove to be a complex problem. The increased wiring between all the qubits can lead to higher likelihood of crosstalk and noise creating unwanted interactions between qubits. This further relies on better error correction algorithms. Though this paper focusses on a theoretical implementation on a quantum simulator, it is still necessary to consider practical implications with NISQ devices.

This architecture provided worse results than the Brick ansatz, often getting stuck at a far higher loss. This implies a difficult loss landscape and the implication of barren plateau. These issue made it unsuitable for use. [ref: https://arxiv.org/pdf/2209.08167]

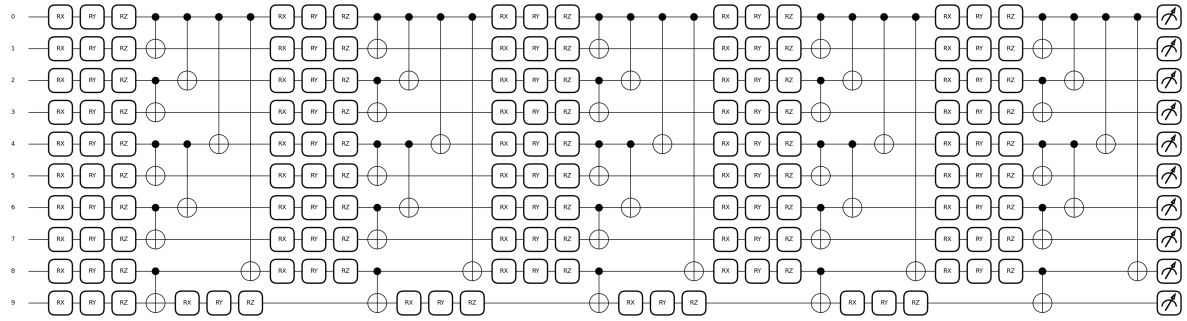

Figure 6: 10x5 Butterfly Ansatz

6.4 ZX Calculus

ZX Calculus is a graphical language designed to express the relationships between qubits based on category theory. Instead of thinking about the linear maps taking place in the form of matrices, we can reason about it in a diagrammatic way. This can make analysing quantum processes easier and more intuitive. This approach makes identifying circuit equalities easier, rather than working out the resultant matrix operation for two given circuits. ZX calculus represents the quantum states and maps as 'spiders' and 'wires'.

At the heart of ZX calculus are two types of generators: Z-spiders and X-spiders, represented by green and red nodes respectively. First looking at the Z-spider:

$$Z_m^n[\alpha] = |0\rangle^{\otimes n}\langle 0|^{\otimes m} + e^{i\alpha}|1\rangle^{\otimes n}\langle 1|^{\otimes m} \tag{6.8}$$

where m is the number of inputs, n is the number of outputs and α is the phase.

X-spiders are similar to Z-spiders but everything is defined in the X-basis rather than the Z-basis.

$$X_m^n[\alpha] = |+\rangle^{\otimes n}\langle +|^{\otimes m} + e^{i\alpha}|-\rangle^{\otimes n}\langle -|^{\otimes m} \tag{6.9}$$

Thinking about the X-spiders in the Z-basis gives us the following:

$$\begin{aligned} |+\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |-\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{aligned} \tag{6.10}$$

and by symmetry

$$\begin{aligned} |0\rangle &= \frac{1}{\sqrt{2}}(|+\rangle + |-\rangle) \\ |1\rangle &= \frac{1}{\sqrt{2}}(|+\rangle - |-\rangle) \end{aligned} \tag{6.11}$$

The graphical rules of ZX calculus allow for spider fusions, copying and wire transformations, all that correspond to those from category theory. It is crucial to note that ZX calculus is universal for quantum computations, so any computation to exist can be represented and reasoned about within this framework.
ADD SPIDER RULES

6.4.1 Gate Optimisation

As mentioned above, non-Clifford gates can be very computationally intensive to implement, therefore we would like to reduce any T-gates that don't need to be in the circuit.

ZX calculus provides us with a framework to optimise the T-gate count by a process referred to as gadgetisation. This involves splitting a non-Clifford node into a node of itself

and a 'phase gadget',a phase that can explore the ZX diagram in search of combination or cancellation.[ref here] The ZX diagrams can then be converted back into a quantum circuit with potentially less non-Clifford gates. This optimisation is implemented in the Python PyZX package[ref here]. I explore the effects of this further in the results section.

7 Results

Putting the theory into practice offered insights into the strengths and weaknesses of the model. This section aims to provide quantitative comparisons between the classical and quantum methods. After careful consideration and analysis of parameters for the QCBM, the model used for comparisons against the MJD model is a 13 qubit, 7 layer model. It uses the brick architecture and has been trained for 500 epochs with an Adam optimiser.

7.1 Path Generation

An important part of risk analysis involves path generation, simulating an equity path for the next n days. This provides a range of final values, aiming to simulate price paths accurately in the process. The metrics involved in the analysis involves comparing: the skewness, excess kurtosis, and standard deviation. For a return r_i and mean return $\bar{r}$
Standard deviation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i - \bar{r})^2}$$

Skewness:

$$\gamma_1 = \frac{\sum_{i=1}^{N}(r_i - \bar{r})^3}{(N-1) \times \sigma^3}$$

Excess kurtosis:

$$\gamma_2 = \frac{1}{N}\frac{\sum_{i=1}^{N}(r_i - \bar{r})^4}{\sigma^4} - 3$$

These were chosen to highlight the accuracy of the models as well as the ability to represent subtleties in the data such as the asymmetric nature and fat tails that are often present in market data. Both models were first calibrated on Brent Crude Oil stock prices. The test period combines the training and unseen data to see how well the model is able to perform on out-of-sample prices. The period of data includes the recent reaction to Trump's tariffs, April 2025. This was included purposely to observe the model's resilience to tough market activity and heightened volatility.

Parameter	**Value**
Drift (μ)	0.37956244
Volatility (σ)	0.40736202
Jump Intensity (λ)	530.63931692
Jump Mean (μ_J)	-0.00374471
Jump Variance (σ_J^2)	0.00065703

Table 1: Calibrated Parameters of the MJD Model for Brent Crude Oil

As shown in table 2, the results highlighted that on training data for Brent Crude Oil, the MJD model was able to capture the standard deviation better, however the skewness and excess kurtosis was represented weakly. This is where the QCBM was able to demonstrate superiority, capturing the skewness and excess kurtosis with greater accuracy.

	Original Data	Test Data	MJD Data	QCBM Data
Standard Deviation	0.0233	0.0225	0.02647	0.0268
Skewness	0.6550	0.6709	-0.2579	0.6215
Excess Kurtosis	8.5986	8.8553	7.4809	8.7375

Table 2: Comparison of Brent data with MJD and QCBM Data

	Original Data	Test Data	MJD Data	QCBM Data
Standard Deviation	0.0120	0.0270	0.0126	0.0124
Skewness	0.8602	1.1640	-0.2015	0.7451
Excess Kurtosis	11.5700	5.6518	11.4330	11.3780

Table 3: Comparison of Eurexx Data with MJD and QCBM Data

Table 3 supports most of the arguments made, though has shown slight superiority in representing the fat tails during the training period. As the Eurexx data contains less extreme jumps, we can see the MJD model perform better. That being said, we can still observe the skewness being learnt poorly by the classical model. The test data chosen is a continuation of the training data, both models showing clear difficulties in generalisation. This can be excused however as the metrics differ hugely from the training data. This does raise a point of concern whether these methods employed of market behaviour prediction are appropriate. For this reason, it is more appropriate to treat the models as a synthetic market generator rather than for the purposes of prediction. The differences in the MJD performance should be noted, highlighting how classical methods tend to struggle in tougher market conditions; this is will be further investigated in the following sections.

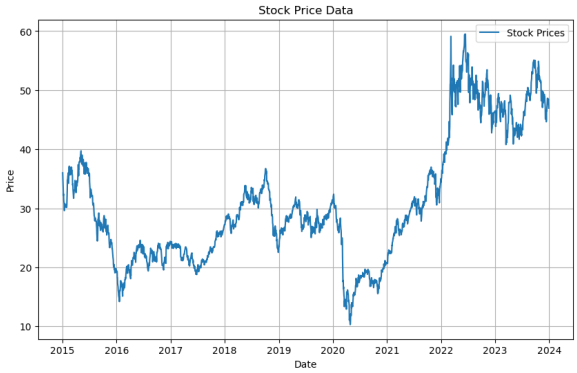

Figure 7: True price path

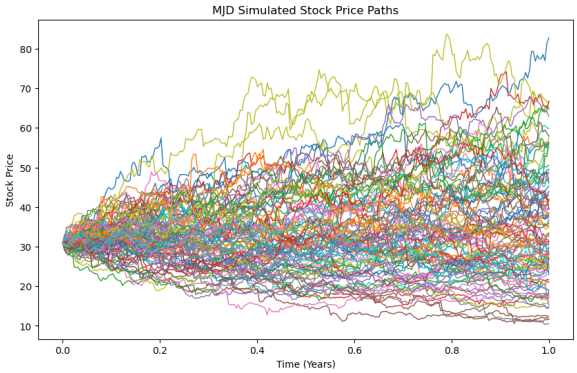

Figure 8: Price trajectories for MJD model

A comparison of the log return distributions in figure 9 indicates the lack of tail representation with the MJD model with QCBM's being visible but with higher density. It is important to note that different parameters for the QCBM gave significantly different results. Adding only 1 more qubit led to an over-estimation in kurtosis as well as poor representation of the standard deviation. From figure 10 we can see behaviour that would on first glance mimic the real market. Jumps in the market look realistic, and trajectories take believable paths. The graph of log returns, figure 11 shows us the heightened log returns in the QCBM with occasional peaks as observed in the market data. This raises

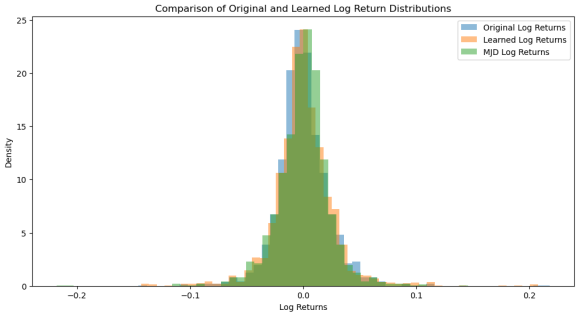

Figure 9: Comparison of distributions

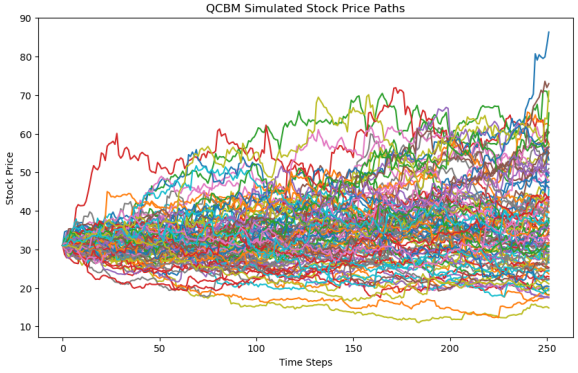

Figure 10: Price trajectories for QCBM model

some concerns on whether this would lead to over-hedging if used in a hedging engine. The cumulative sum of returns shows a similarity in the global returns we would expect, the MJD in this regard performs weakly, indicating that the market returned higher returns, an assumption that may lead to under-hedging, leaving an investor exposed to more risk than calculated.

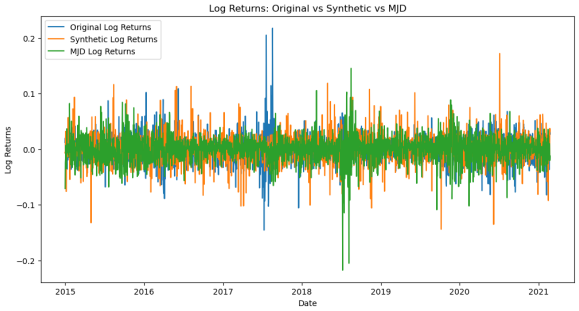

Figure 11: Log returns

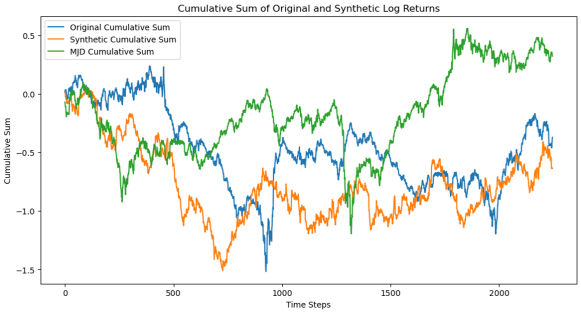

Figure 12: Cumulative sum of returns

We can also compare the ACF(Autocorrelation Function) values to see how well the QCBM has replicated the structure of the original dataset. ACF aims to quantify the correlation between observations separated by a lag value, k. This will give us insight into market microstructures within the original data as well as seeing if the QCBM is able to learn these complexities as well. Formally the ACF for a lag k and time series X, can be defined as:

$$\rho_k = \frac{Cov(X_t, X_{t-k})}{\sqrt{Var(X_t) \cdot Var(X_{t-k})}} \tag{7.1}$$

Plots 13 and 14 show the structure of the training data vs the QCBM data. The market data shows significant correlation between the lags, indicating volatility clustering. This is not reflected in the QCBM data, with random spikes showing the model struggles with representing volatility persistence. Similar behaviour is observed in the MJD model as shown in figure 15. We continue this analysis in further detail in the next section.

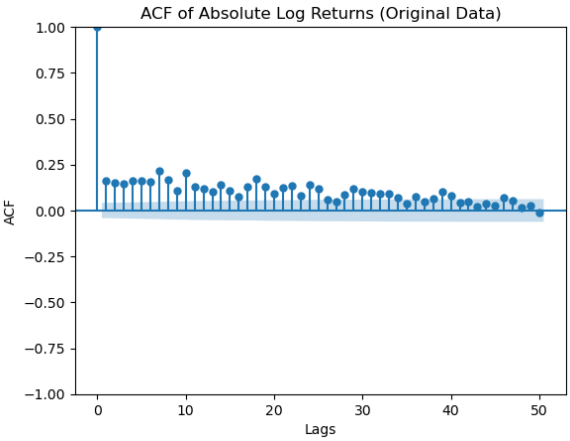

Figure 13: ACF plot of market data

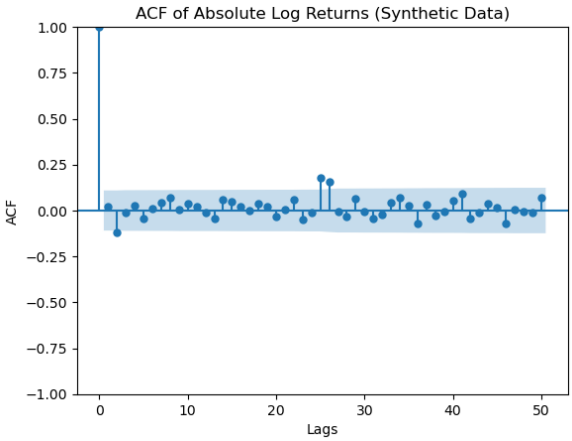

Figure 14: ACF plot of QCBM data

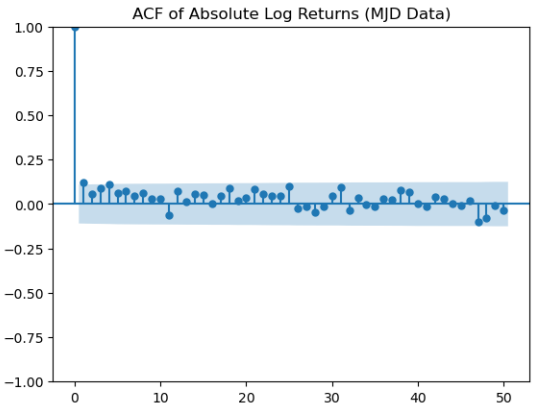

Figure 15: ACF plot of MJD data

7.2 Volatility Analysis

It is also important to analyse if the local behaviour of the equity paths is well captured. We can do this by analysing the volatility; here the shortfalls of the QCBM become clear. Though the global volatility distribution appears to well represented, using GARCH models shows a disparity in observed volatility clustering compared to the quantum generated volatility. This appears to be a fundamental flaw in the model. The assumption that we can represent a path using [give equation for propagating path] leads to poor volatility clustering as we have treated for each time $0 \leq t_0 \leq ... \leq t_n$ our random variable $S_{t_r} - S_{t_{r-1}}$ are independent.

First looking at rolling volatility, we observe acceptable performance by the MJD model, displaying realistic market volatility; the QCBM, however, displayed a worse performance, often underestimating volatility peaks, and overestimating noise as shown in figure 16. This heightened noise is also reflected in figure 17 where we see on a weekly basis, the QCBM constantly overestimated the volatility. We can observe the difference in learnt volatility by comparing distributions. Figure 18 shows us how the volatility is skewed with a larger mean and lack of kurtosis. This story worsens as we look at using volatility

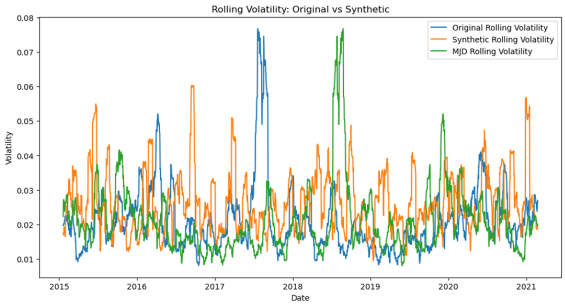

Figure 16: Comparison of rolling volatility (20-day window)

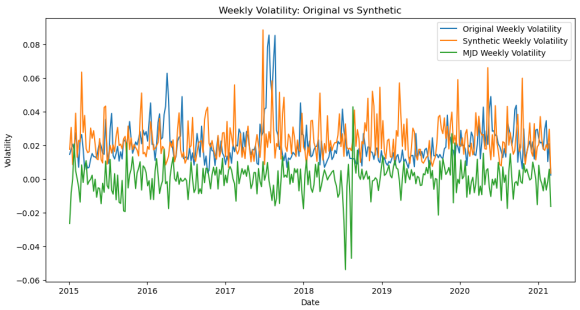

Figure 17: Comparison of weekly volatility

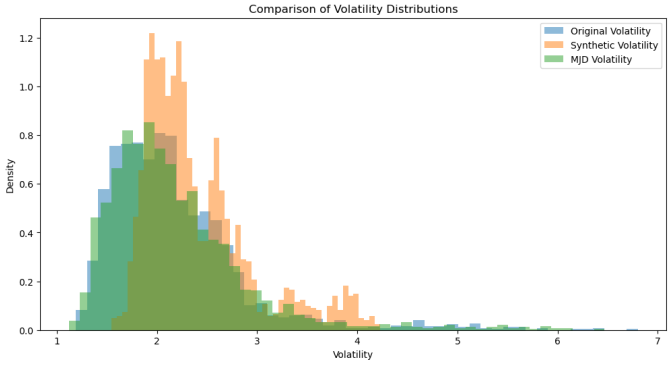

Figure 18: Comparison of volatility distributions

models as a comparison method.

GARCH (Generalised Autoregressive Conditional Heteroskedasticity) models were first introduced in 1982 by Robert Eagle [add reference here] to model volatility as a non-constant quantity in financial models. The GARCH(1,1) can be defined as so:
Let r_t be the asset return at time t. This can be decomposed as

$$r_t = \mu + \epsilon_t \tag{7.2}$$

where μ is the mean return and $\epsilon_t = \sigma_t z_t$ where $z \sim N(0,1)$. GARCH models the conditional variance of a given time series process, asset returns in our world, as a function of past squared shocks (ϵ_{t-1}^2) and past conditional variance (σ_{t-1}^2). Using this, our model equation becomes

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2 \tag{7.3}$$

where $\omega > 0$ is the average volatility, $\alpha \geq 0$ is the sensitivity to recent shocks, and $\beta \geq 0$ is the persistence of volatility, the tendency of volatility to be high for periods of time and then low of periods of time.

A further model that improved on GARCH is the Exponential GARCH, another model used for comparison in my findings. This model focussed on asymmetric volatility effects, removing parameter restrictions, and having a logarithmic formulation. We can define an

EGARCH(1,1) as follows

$$\ln(\sigma_t^2) = \omega + \beta \ln(\sigma_{t-1}^2) + \gamma \frac{\epsilon_{t-1}}{\sigma_{t-1}} + \alpha \left(\frac{|\epsilon_{t-1}|}{\sigma_{t-1}} - \sqrt{\frac{2}{\pi}} \right) \tag{7.4}$$

where the extra term γ accounts for the leverage effect; the negative correlation between asset returns and volatility change often observed in markets.

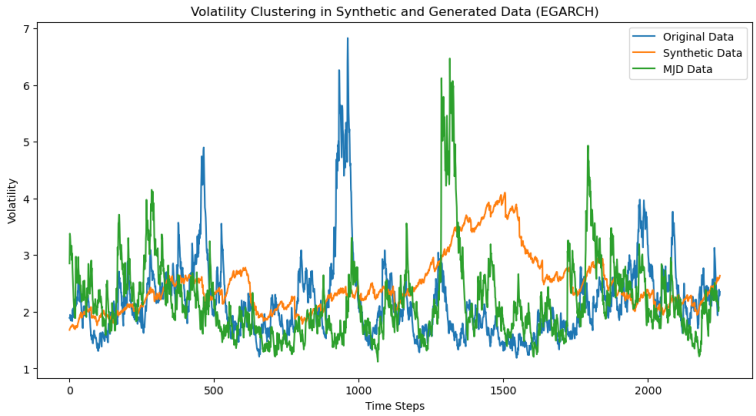

Figure 19: EGARCH model fit to models

The implementation and parameter estimation for ω,α, β and γ were all handled by the python package 'arch' [insert this reference https://pypi.org/project/arch/]. Calibrating a EGARCH(1,1) to the different model returns gave the following graph. In figure 19 we can observe the MJD model displaying more accurate clustering compared to the QCBM which remains conservative. This may suggest that the MJD model is more suitable for predicting volatility over a short period of time. As an improvement to the QCBM, it may be of use to create a hybrid model, one that combines the QCBM and a volatility model of choice.

7.3 VaR & CVaR

To determine the usefulness of the generator for hedging purposes, we must explore how both generators perform in the extreme percentiles. These are the scenarios that we call Black-Swan events; an event that is unexpected, infrequent but has huge financial implications. Not being hedged against such changes can leave an investor in unwanted positions, exposed to large changes and at risk of losing a lot of money. To quantify expected loss a market may give, we employ techniques such as VaR(Value at Risk) and CVaR(Conditional Value at Risk).

VaR is a measure that focuses on the loss at a given percentile under normal market conditions. It provides a threshold such that the probability of a loss exceeding a given value is a chosen percentile i.e 1% or 5%. An intuitive way to think about it is 1% VaR of -0.05 means there is a 1% chance of losing at least 5% of the portfolio value. Mathematically we can define VaR:
Let X be a random variable with cdf $F_X(z) = P\{X \leq z\}$, this is our loss function. The VaR of X at a confidence level α is given by

$$VaR_\alpha(X) = \min\{z | F_X(z) \geq \alpha\} \tag{7.5}$$

VaR, however, has a few crucial limitations. VaR does not say anything about the size of losses made after the given α value. This makes it very hard to quantify the loss an investor may be exposed to. VaR also assumes liquidity in the market at any position when in reality during periods of market stress, the ability to buy or sell at any price can be very difficult. It should also be known that VaR has a non-subadditivity constraint, meaning VaR for different portfolios cannot be added together. For these reasons we opt to use Conditional Value at Risk, a more sophisticated tool designed to overcome the given limitations.

CVaR focusses on measuring the risk of extreme losses. Instead of considering the probability of losing a given amount, we instead focus on how much is lost in the given quantile. This quantifies our tail risk, making it appropriate for evaluating the heavy tails of our market generators. Intuitively a 1% CVaR of -0.05 means that, in the worst 1% of cases, the average loss is 5%. We can define CVaR at a level α as

$$CVaR_\alpha(X) = \mathbb{E}[X | X \geq VaR_\alpha(X)] \tag{7.6}$$

This provides us with a sturdier framework to evaluate performances although we must still be careful of its assumptions and limitations. We must assume that the distribution of the loss is measured accurately and market conditions are also represented faithfully within the given time window. Once we accept these to be true, we are provided with a method to compare strategies and generators, as well as conforming to financial regulations such as Basel III.

The first test was on Eurexx data from the COVID period. This was excluded from the training dataset to be used as an extreme market situation. Market generators need to be resilient and able to account such market shocks, these are the scenarios where hedging

	VaR (1%)	**CVaR (1%)**
Original Data	-0.0171	-0.0174
QCBM Simulated	-0.0291	-0.0308
MJDM Simulated	-0.0360	-0.0377

Table 4: VaR & CVaR at the 1% Quantile for COVID data

	VaR (5%)	**CVaR (5%)**
Original Data	-0.0159	-0.0167
QCBM Simulated	-0.0223	-0.0270
MJDM Simulated	-0.0283	-0.0338

Table 5: VaR & CVaR at the 5% Quantile for COVID data

positions stop investors from losing large amounts of money. At the 1% quantile we can see both the MJD and QCBM exhibit greater losses in the tails compared to the real data. The QCBM has a VaR of 2.91% and CVaR of 3.08%, these are both significant overestimates of tail risk present in the training data. The MJD overestimates the tail risk even greater, expecting much larger than expected losses. We could explain this due to the jump component. These pessimistic scenarios, though stopping large losses, in reality may lead to over hedging, a situation where a given investor may be too protected to the market, limiting potential gains. These scenarios may be suboptimal for a deep hedging engine, where the optimal hedge is going to be larger than needed.

At the 5% quantile we see a similar story fold out, both generators estimating the tail risk to be greater than observed. The QCBM once again provides a closer score to the data, indicating the model is more suitable for hedging purposes, though only marginally. What should be noted is the large gap between the VaR and CVaR, indicating the MJD model has accounted for significant market shocks, indicating high sensitivity to extreme events.

The next scenario is on the unseen continuation of the Brent Crude Oil dataset, though not excluded purposely than just for out-of-sample testing, the market conditions observed have been very volatile. As mentioned earlier, the effect of Trump's tariffs were felt in all markets, particularly affecting assets denominated in the US dollar. Though as unfortunate as the scenario may be to investors, this has provided me with an excellent test for the two models.

	VaR (1%)	**CVaR (1%)**
Original Data	-0.0300	-0.0342
QCBM Simulated	-0.0429	-0.0524
MJDM Simulated	-0.0522	-0.0631

Table 6: VaR & CVaR at the 1% Quantile for Brent Crude Data

	VaR (5%)	**CVaR (5%)**
Original Data	-0.0239	-0.0274
QCBM Simulated	-0.0313	-0.0391
MJDM Simulated	-0.0414	-0.0508

Table 7: VaR & CVaR at the 5% Quantile for Brent Crude Data

Though the data was more volatile, both models demonstrated that they are able to capture the tail risk. The training dataset had large amounts of volatility so we would expect good representation of tail risk. As seen with the COVID results, the models are overestimating the downside risk with scenarios showing greater loss than expected. The large gap between the VaR and CVaR values once again tells us that the underlying models predict tail events far higher than observed in the market. It should be said that the QCBM model is able to reflect the tail risk more accurately than the MJD, with value that are closer to the original data. One might make conclusions that this makes the QCBM a useful model but the poor performance to the original data raises concerns; we can however say that the model is more appropriate to use than the MJD in this scenario.

7.4 Barren Plateau

Upon training it was evident that the circuits used: Brick, Butterfly, and Pyramid included, all had issues with trainability; the loss function would converge at suboptimal parameters. This led me to investigate the loss landscape and reason about the difficulties with training the circuit.

We first have to confirm whether the barren plateau(BP) is present. As discussed earlier, BP occurs when the gradient of the cost function with respect to the parameters, $\frac{\partial C}{\partial \theta}$, tends to 0. Plotting this gave us results that were unexpected. Figure 20 shows us that the variance of the gradient increases as we increase the qubit count, going against what the maths would suggest. One theory was that the nature of the data led to such phenomenon.

To confirm such beliefs, I used the QCBM to learn several distributions. These included:

- Step function: $H(x) = \begin{cases} 0 & \text{if } x < 0, \\ 1 & \text{if } x \geq 0 \end{cases}$
- Dirac delta function: $\delta(x) = \begin{cases} \infty & \text{if } x = 0, \\ 0 & \text{otherwise} \end{cases}$ with $\int_{-\infty}^{\infty} \delta(x)\, dx = 1$
- Uniform distribution: $U(x; a, b) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq x \leq b, \\ 0 & \text{otherwise} \end{cases}$
- Normal distribution: $\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

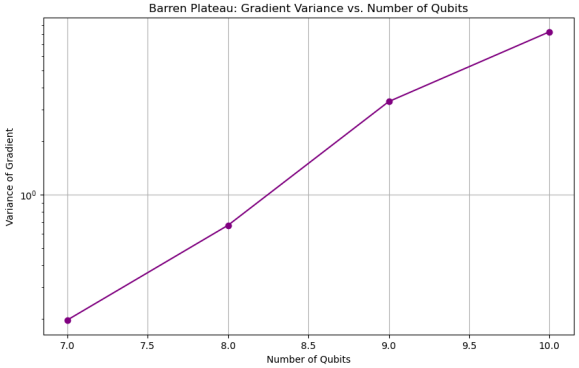

Figure 20: Variance of the gradient vs qubit count

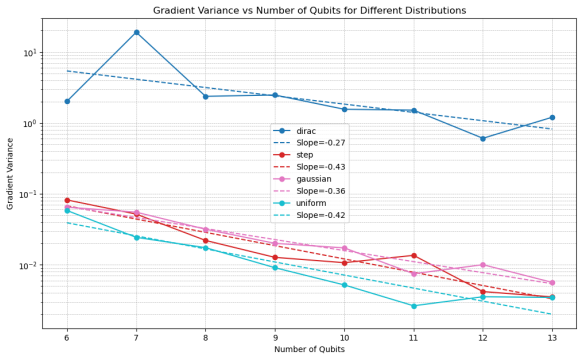

Figure 21: Variance of gradients vs qubit count for multiple distributions

What is intriguing is how the dirac function performed. This suggested spikes in the data had an impact on the trainability, to further reinforce this point I tested how the QCBM performed with a function that contains many spikes. Figures 22 & 23 show the

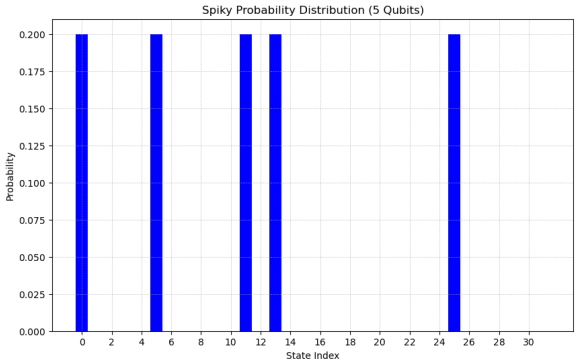

Figure 22: Spiky distribution

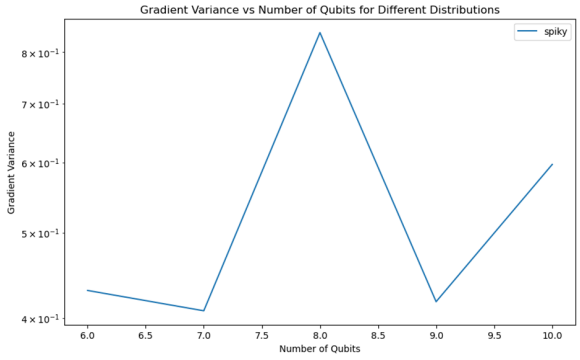

Figure 23: Variance of gradients vs qubit count for a spiky distribution

distribution learnt and how the variance of the cost gradients change. The spiky nature of the plot indicates that there is something else affecting the QCBM's performance beyond just barren plateau. This effect was beyond the scope of this report but can be investigated after this project.

7.4.1 ZX Calculus

Reducing the T-gate count was essential to improve the trainability. I first plotted the loss landscape for some of the gate combinations to visualise how the loss changes with different gate pairs. As shown in the graphs, the optimisation process for the first layers is much simpler to converge than for the latter gates. The local minimas presented in comparing layers 4 and 6 demonstrates the need to reduce gate count to simplify the optimisation process. The first attempt at T-gate optimisation included rounding all the θ values for the PQC to the nearest $\frac{\pi}{8}$. This process allows us to use the ZX-calculus reduction rules to reduce the T-gate count. Though this reduced the number of non-

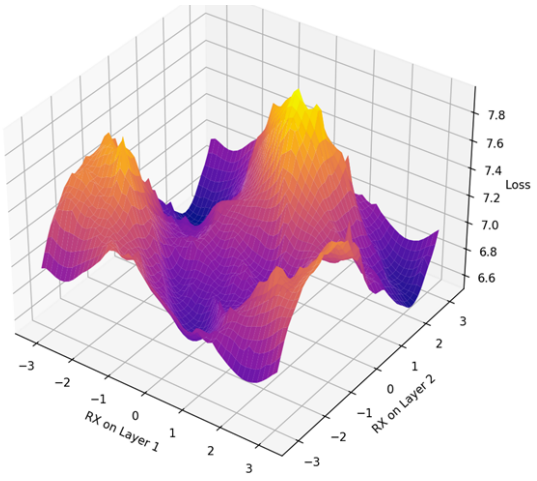

Figure 24: Loss landscape for gates in the first 2 layers

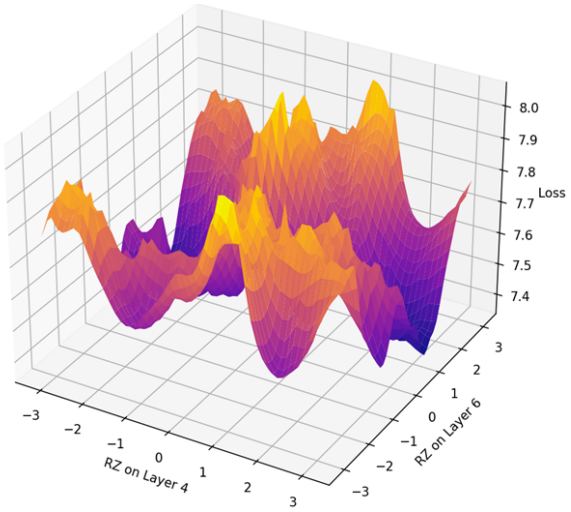

Figure 25: Loss landscape for gates at an increased depth

Clifford gates in the circuit, the performance took a large hit. There was a significant reduction in the standard deviation and excess kurtosis approximation. The loss landscape also indicated that first layers had less of an effect on the QCBM performance so I tested only rounding the first two layers. This led to better results with standard deviation being less affected though the excess kurtosis was represented poorly. This trade-off is not acceptable from a hedging standpoint as it would lead to a suboptimal hedge, leaving an investor over-exposed or incurring more transaction costs.

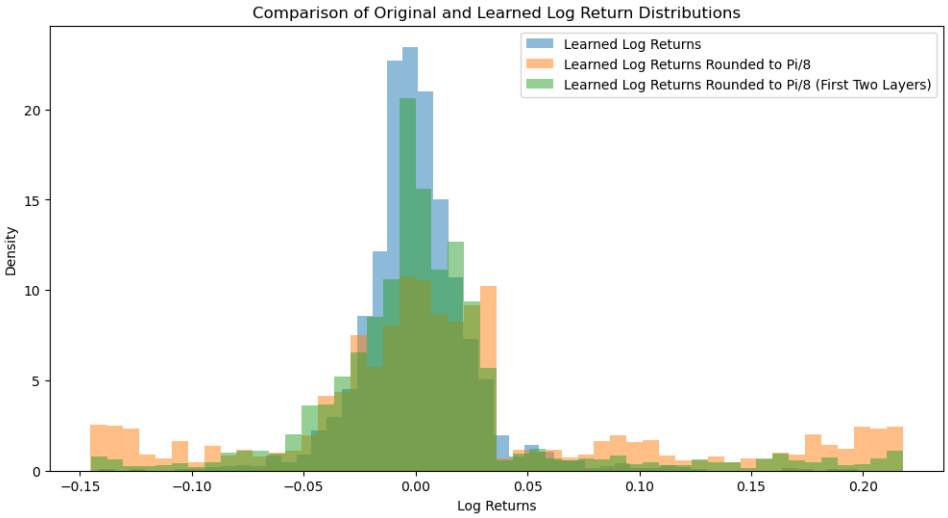

Figure 26: Comparison of learned distributions

8 Project Management

Gantt chart and brief explanation here.

9 Outlook and Conclusions

This project was designed to investigate the quantum advantages as compared to traditional techniques for deep hedging, and all the goals set out have been fulfilled. Through thorough analysis of model performances, I am able to say that the QCBM outperformed the Merton-Jump Diffusion model in modelling the fat tails and skewness of the market data.

In the context of hedging, the synthetic data generated would most likely not be acceptable as the lack of volatility clustering may lead to suboptimal hedges in the hedging engine. The overestimation of tail risk was also prevalent, possibly leading to an over-hedge, minimising the potential upside of a trade. It should also be said, while talking to oil traders, it came up that traders would often under hedge their positions. This is due to the fact they are placing a directional bet based on their reasoning, be it mathematical or due to experience. This is a factor that is not considered but should be thought about when designing a hedging engine. This further emphasising the point that though the QCBM appears superior in the chosen metrics, it may not be completely suitable for a hedging engine. For further analysis, comparisons against machine learning techniques can be executed, deducing whether this method beats all methods of simulation on classical devices.

I also explored issues with trainability, coming across a phenomenon where the variance of the cost gradient increased, or appeared uncorrelated when dealing with distributions with scattered high densities. Beyond this research, I would like to explore this, mathematically reasoning the effect and mitigation techniques.

Optimisation techniques such as T-gate reduction using ZX calculus was also explored, showing the strength of such representation. I believe this another area of research that can be extended beyond this project, investigating different gate reduction techniques, ones perhaps that may be better suited for this nature of data.

There is new research being done on Quantum Monte Carlo techniques, boasting superiority over classical quasi-monte carlo methods due to the truly random nature of quantum mechanics. Moving beyond this paper, this technique can be added easily to the current framework for simulation of asset prices and deep hedging.

This technique of generating synthetic data has applications beyond deep hedging and quantitative finance. The framework set up can be adapted for use cases where lack of training data is present, or for scenarios where the underlying distribution is difficult to learn classically.

10 Appendix

10.1 QCBM Architectures

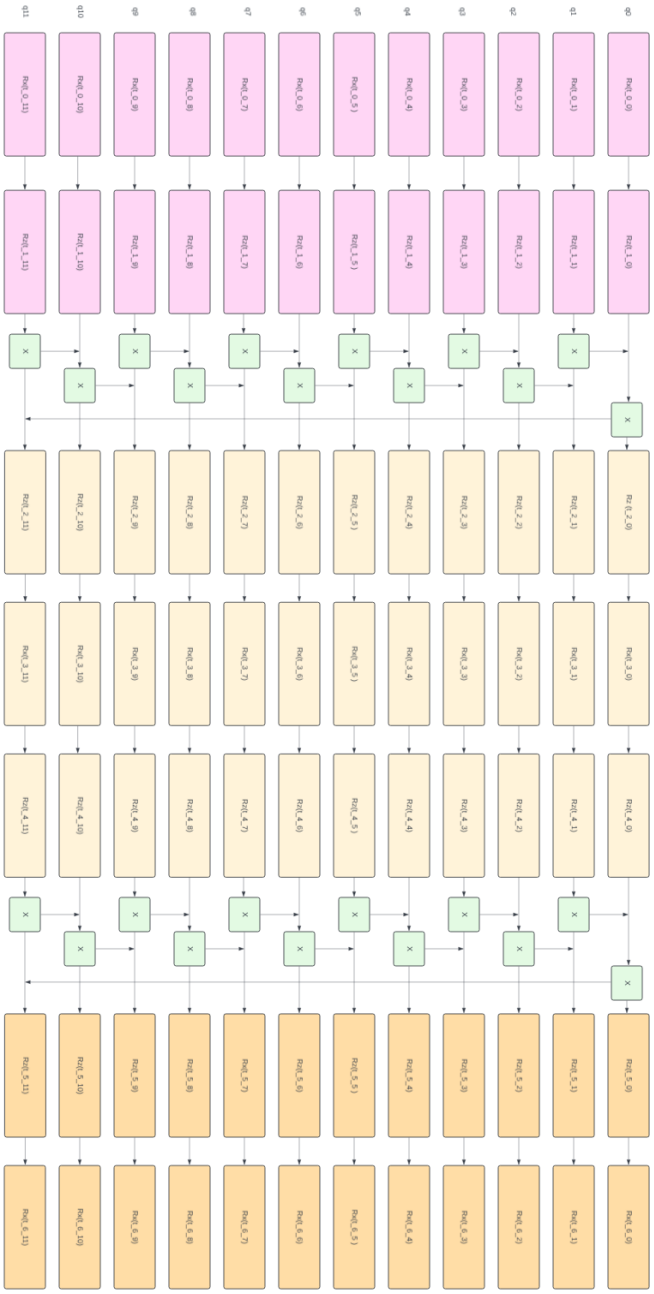

Figure 27: QCBM architecture

References

[1] John Armstrong and George Tatlow. *Deep Gamma Hedging*. Sept. 20, 2024. DOI: 10.48550/arXiv.2409.13567. arXiv: 2409.13567[q-fin]. URL: http://arxiv.org/abs/2409.13567 (visited on 12/04/2024).

[2] Marcello Benedetti et al. "A generative modeling approach for benchmarking and training shallow quantum circuits". In: *npj Quantum Information* 5.1 (May 27, 2019), p. 45. ISSN: 2056-6387. DOI: 10.1038/s41534-019-0157-8. URL: https://www.nature.com/articles/s41534-019-0157-8 (visited on 11/21/2024).

[3] Marcello Benedetti et al. "Erratum: Parameterized quantum circuits as machine learning models (2019 *Quant. Sci. Tech.* **4** 043001)". In: *Quantum Science and Technology* 5.1 (Dec. 4, 2019), p. 019601. ISSN: 2058-9565. DOI: 10.1088/2058-9565/ab5944. URL: https://iopscience.iop.org/article/10.1088/2058-9565/ab5944 (visited on 11/21/2024).

[4] Michael R. Berthold, Ad Feelders, and Georg Krempl, eds. *Advances in Intelligent Data Analysis XVIII: 18th International Symposium on Intelligent Data Analysis, IDA 2020, Konstanz, Germany, April 27–29, 2020, Proceedings*. Vol. 12080. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020. ISBN: 978-3-030-44583-6 978-3-030-44584-3. DOI: 10.1007/978-3-030-44584-3. URL: http://link.springer.com/10.1007/978-3-030-44584-3 (visited on 12/04/2024).

[5] Bloomberg. *Bloomberg*. Bloomberg.com, 2023. URL: https://www.bloomberg.com/.

[6] Nicolas Boursin, Carl Remlinger, and Joseph Mikael. "Deep Generators on Commodity Markets Application to Deep Hedging". In: *Risks* 11.1 (Dec. 23, 2022), p. 7. ISSN: 2227-9091. DOI: 10.3390/risks11010007. URL: https://www.mdpi.com/2227-9091/11/1/7 (visited on 11/21/2024).

[7] Hans Buehler, Murray Phillip, and Ben Wood. "Deep Bellman Hedging". In: *SSRN Electronic Journal* (2022). ISSN: 1556-5068. DOI: 10.2139/ssrn.4151026. URL: https://www.ssrn.com/abstract=4151026 (visited on 11/21/2024).

[8] Hans Buehler et al. "Deep Hedging: Hedging Derivatives Under Generic Market Frictions Using Reinforcement Learning". In: *SSRN Electronic Journal* (2019). ISSN: 1556-5068. DOI: 10.2139/ssrn.3355706. URL: https://www.ssrn.com/abstract=3355706 (visited on 11/21/2024).

[9] El Amine Cherrat et al. "Quantum Deep Hedging". In: *Quantum* 7 (Nov. 29, 2023), p. 1191. ISSN: 2521-327X. DOI: 10.22331/q-2023-11-29-1191. arXiv: 2303.16585[quant-ph]. URL: http://arxiv.org/abs/2303.16585 (visited on 12/04/2024).

[10] Vedant Choudhary, Sebastian Jaimungal, and Maxime Bergeron. *FuNVol: A Multi-Asset Implied Volatility Market Simulator using Functional Principal Components and Neural SDEs*. Dec. 26, 2023. DOI: 10.48550/arXiv.2303.00859. arXiv: 2303.00859[q-fin]. URL: http://arxiv.org/abs/2303.00859 (visited on 12/04/2024).

[11] Eliahu Cohen. "What Weak Measurements and Weak Values Really Mean: Reply to Kastner". In: *Foundations of Physics* 47.10 (Oct. 2017), pp. 1261–1266. ISSN: 0015-9018, 1572-9516. DOI: 10.1007/s10701-017-0107-2. URL: http://link.springer.com/10.1007/s10701-017-0107-2 (visited on 11/21/2024).

[12] Kalyan Dasgupta and Binoy Paine. *Loading Probability Distributions in a Quantum circuit*. Aug. 29, 2022. arXiv: 2208.13372[quant-ph]. URL: http://arxiv.org/abs/2208.13372 (visited on 11/21/2024).

[13] Julien Dudas et al. "Quantum reservoir computing implementation on coherently coupled quantum oscillators". In: *npj Quantum Information* 9.1 (July 7, 2023), p. 64. ISSN: 2056-6387. DOI: 10.1038/s41534-023-00734-4. URL: https://www.nature.com/articles/s41534-023-00734-4 (visited on 11/21/2024).

[14] *EURO STOXX 50*. Wikipedia, Aug. 2021. URL: https://en.wikipedia.org/wiki/EURO_STOXX_50.

[15] Simon Fecamp, Joseph Mikael, and Xavier Warin. "Deep learning for discrete-time hedging in incomplete markets". In: *The Journal of Computational Finance* (2021). ISSN: 14601559, 17552850. DOI: 10.21314/JCF.2021.006. URL: https://www.risk.net/journal-of-computational-finance/7871526/deep-learning-for-discrete-time-hedging-in-incomplete-markets (visited on 12/04/2024).

[16] Giacomo Franceschetto et al. *Harnessing quantum back-action for time-series processing*. Nov. 6, 2024. arXiv: 2411.03979[quant-ph]. URL: http://arxiv.org/abs/2411.03979 (visited on 11/21/2024).

[17] Keisuke Fujii and Kohei Nakajima. *Quantum reservoir computing: a reservoir approach toward quantum machine learning on near-term quantum devices*. Nov. 10, 2020. arXiv: 2011.04890[quant-ph]. URL: http://arxiv.org/abs/2011.04890 (visited on 11/21/2024).

[18] Santanu Ganguly. "Implementing Quantum Generative Adversarial Network (qGAN) and QCBM in Finance". In: ().

[19] Jorge García-Beni et al. "Squeezing as a resource for time series processing in quantum reservoir computing". In: *Optics Express* 32.4 (Feb. 12, 2024), p. 6733. ISSN: 1094-4087. DOI: 10.1364/OE.507684. arXiv: 2310.07406[quant-ph]. URL: http://arxiv.org/abs/2310.07406 (visited on 11/21/2024).

[20] Kaitlin Gili et al. *Do Quantum Circuit Born Machines Generalize?* arXiv.org, 2022. URL: https://arxiv.org/abs/2207.13645.

[21] L. C. G. Govia et al. "Quantum reservoir computing with a single nonlinear oscillator". In: *Physical Review Research* 3.1 (Jan. 25, 2021), p. 013077. ISSN: 2643-1564. DOI: 10.1103/PhysRevResearch.3.013077. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.3.013077 (visited on 11/21/2024).

[22] Haim Horowitz, Pooja Rao, and Santosh Kumar Radha. *A quantum generative model for multi-dimensional time series using Hamiltonian learning*. Apr. 13, 2022. DOI: 10.48550/arXiv.2204.06150. arXiv: 2204.06150[quant-ph]. URL: http://arxiv.org/abs/2204.06150 (visited on 12/04/2024).

[23] Iordanis Kerenidis, Jonas Landman, and Natansh Mathur. *Classical and Quantum Algorithms for Orthogonal Neural Networks*. Dec. 23, 2022. arXiv: 2106.07198[quant-ph]. URL: http://arxiv.org/abs/2106.07198 (visited on 11/21/2024).

[24] Alex Kondratyev. "Non-Differentiable Leaning of Quantum Circuit Born Machine with Genetic Algorithm". In: *Wilmott* 2021.114 (July 2021), pp. 50–61. ISSN: 1540-6962, 1541-8286. DOI: 10.1002/wilm.10943. URL: https://onlinelibrary.wiley.com/doi/10.1002/wilm.10943 (visited on 11/21/2024).

[25] Jin-Guo Liu and Lei Wang. "Differentiable Learning of Quantum Circuit Born Machine". In: *Physical Review A* 98.6 (Dec. 19, 2018), p. 062324. ISSN: 2469-9926, 2469-9934. DOI: 10.1103/PhysRevA.98.062324. arXiv: 1804.04168[quant-ph]. URL: http://arxiv.org/abs/1804.04168 (visited on 11/21/2024).

[26] *LSEG Workspace*. www.lseg.com. URL: https://www.lseg.com/en/data-analytics/products/workspace.

[27] Ali Asghar Movahed and Houshyar Noshad. "Introducing a new approach for modeling a given time series based on attributing any random variation to a jump event: jump-jump modeling". In: *Scientific Reports* 14.1 (Jan. 12, 2024), p. 1234. ISSN: 2045-2322. DOI: 10.1038/s41598-024-51863-5. URL: https://www.nature.com/articles/s41598-024-51863-5 (visited on 11/21/2024).

[28] Pere Mujal et al. "Time-series quantum reservoir computing with weak and projective measurements". In: *npj Quantum Information* 9.1 (Feb. 23, 2023), p. 16. ISSN: 2056-6387. DOI: 10.1038/s41534-023-00682-z. URL: https://www.nature.com/articles/s41534-023-00682-z (visited on 11/21/2024).

[29] Bernt Øksendal. "Application to Mathematical Finance". In: *Stochastic Differential Equations: An Introduction with Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 269–313. ISBN: 978-3-642-14394-6. DOI: 10.1007/978-3-642-14394-6_12. URL: https://doi.org/10.1007/978-3-642-14394-6_12.

[30] Anupama Padha and Anita Sahoo. "Quantum deep neural networks for time series analysis". In: *Quantum Information Processing* 23.6 (May 24, 2024), p. 205. ISSN: 1573-1332. DOI: 10.1007/s11128-024-04404-y. URL: https://link.springer.com/10.1007/s11128-024-04404-y (visited on 11/21/2024).

[31] Annie E. Paine, Vincent E. Elfving, and Oleksandr Kyriienko. "Quantum Quantile Mechanics: Solving Stochastic Differential Equations for Generating Time-Series". In: *Advanced Quantum Technologies* 6.10 (Oct. 2023), p. 2300065. ISSN: 2511-9044, 2511-9044. DOI: 10.1002/qute.202300065. arXiv: 2108.03190[quant-ph]. URL: http://arxiv.org/abs/2108.03190 (visited on 11/21/2024).

[32] Obdulia Pichardo Lagunas, Juan Martínez-Miranda, and Bella Martínez Seis, eds. *Advances in Computational Intelligence: 21st Mexican International Conference on Artificial Intelligence, MICAI 2022, Monterrey, Mexico, October 24–29, 2022, Proceedings, Part I*. Vol. 13612. Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, 2022. ISBN: 978-3-031-19492-4 978-3-031-19493-1. DOI: 10.1007/978-3-031-19493-1. URL: https://link.springer.com/10.1007/978-3-031-19493-1 (visited on 11/21/2024).

[33] Chunhui Qiao and Xiangwei Wan. *Enhancing Black-Scholes Delta Hedging via Deep Learning*. Aug. 24, 2024. DOI: 10.48550/arXiv.2407.19367. arXiv: 2407.19367[q-fin]. URL: http://arxiv.org/abs/2407.19367 (visited on 12/04/2024).

[34] Lupei Qin, Wei Feng, and Xin-Qi Li. "Simple understanding of quantum weak values". In: *Scientific Reports* 6.1 (Feb. 3, 2016), p. 20286. ISSN: 2045-2322. DOI: 10.1038/srep20286. URL: https://www.nature.com/articles/srep20286 (visited on 11/21/2024).

[35] Michael Ragone et al. "A Lie algebraic theory of barren plateaus for deep parameterized quantum circuits". In: *Nature Communications* 15 (Aug. 2024). DOI: 10.1038/s41467-024-49909-3. (Visited on 11/19/2024).

[36] Samir Saissi Hassani and Georges Dionne. "The New International Regulation of Market Risk: Roles of VaR and CVaR in Model Validation". In: *SSRN Electronic Journal* (2021). ISSN: 1556-5068. DOI: 10.2139/ssrn.3766511. URL: https://www.ssrn.com/abstract=3766511 (visited on 12/04/2024).

[37] Samir Saissi Hassani and Georges Dionne. "The New International Regulation of Market Risk: Roles of VaR and CVaR in Model Validation". In: *SSRN Electronic Journal* (2021). DOI: 10.2139/ssrn.3766511. (Visited on 01/16/2022).

[38] Martin Schweizer and Johannes Wissel. "Arbitrage-free market models for option prices: the multi-strike case". In: *Finance and Stochastics* 12.4 (Oct. 2008), pp. 469–505. ISSN: 0949-2984, 1432-1122. DOI: 10.1007/s00780-008-0068-6. URL: http://link.springer.com/10.1007/s00780-008-0068-6 (visited on 12/04/2024).

[39] Ali Shaib et al. "Efficient noise mitigation technique for quantum computing". In: *Scientific Reports* 13.1 (Mar. 8, 2023), p. 3912. ISSN: 2045-2322. DOI: 10.1038/s41598-023-30510-5. URL: https://www.nature.com/articles/s41598-023-30510-5 (visited on 11/21/2024).

[40] Lina Song. "Dynamic Modeling and Simulation of Option Pricing Based on Fractional Diffusion Equations with Double Derivatives". In: *Computational Economics* (May 26, 2024). ISSN: 0927-7099, 1572-9974. DOI: 10.1007/s10614-024-10628-y. URL: https://link.springer.com/10.1007/s10614-024-10628-y (visited on 12/04/2024).

[41] Apimuk Sornsaeng, Ninnat Dangniam, and Thiparat Chotibut. "Quantum next generation reservoir computing: an efficient quantum algorithm for forecasting quantum dynamics". In: *Quantum Machine Intelligence* 6.2 (Dec. 2024), p. 57. ISSN: 2524-4906, 2524-4914. DOI: 10.1007/s42484-024-00188-7. URL: https://link.springer.com/10.1007/s42484-024-00188-7 (visited on 11/21/2024).

[42] Nikitas Stamatopoulos and William J. Zeng. "Derivative Pricing using Quantum Signal Processing". In: *Quantum* 8 (Apr. 30, 2024), p. 1322. ISSN: 2521-327X. DOI: 10.22331/q-2024-04-30-1322. arXiv: 2307.14310[quant-ph]. URL: http://arxiv.org/abs/2307.14310 (visited on 11/21/2024).

[43] György Steinbrecher and William T. Shaw. "Quantile mechanics". In: *European Journal of Applied Mathematics* 19.2 (Apr. 2008). ISSN: 0956-7925, 1469-4425. DOI: 10.1017/S0956792508007341. URL: http://www.journals.cambridge.org/abstract_S0956792508007341 (visited on 12/04/2024).

[44] Andrew Vlasic. *Quantum Circuits, Feature Maps, and Expanded Pseudo-Entropy: A Categorical Theoretic Analysis of Encoding Real-World Data into a Quantum Computer*. Oct. 29, 2024. DOI: 10.48550/arXiv.2410.22084. arXiv: 2410.22084[quant-ph]. URL: http://arxiv.org/abs/2410.22084 (visited on 12/04/2024).

[45] Magnus Wiese et al. "Deep Hedging: Learning to Simulate Equity Option Markets". In: *SSRN Electronic Journal* (2019). ISSN: 1556-5068. DOI: 10.2139/ssrn.3470756. URL: https://www.ssrn.com/abstract=3470756 (visited on 11/21/2024).

[46] Magnus Wiese et al. *Multi-Asset Spot and Option Market Simulation*. Dec. 13, 2021. DOI: 10.48550/arXiv.2112.06823. arXiv: 2112.06823[q-fin]. URL: http://arxiv.org/abs/2112.06823 (visited on 12/04/2024).

[47] Neil A. Wilmot and Charles F. Mason. "Jump Processes in the Market for Crude Oil". In: *The Energy Journal* 34.1 (Jan. 2013), pp. 33–48. ISSN: 0195-6574, 1944-9089. DOI: 10.5547/01956574.34.1.2. URL: https://journals.sagepub.com/doi/10.5547/01956574.34.1.2 (visited on 11/21/2024).

[48] Xiao-Ming Zhang, Tongyang Li, and Xiao Yuan. "Quantum State Preparation with Optimal Circuit Depth: Implementations and Applications". In: *Physical Review Letters* 129.23 (Nov. 30, 2022), p. 230504. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.129.230504. arXiv: 2201.11495[quant-ph]. URL: http://arxiv.org/abs/2201.11495 (visited on 11/21/2024).

[49] Christa Zoufal, Aurélien Lucchi, and Stefan Woerner. "Quantum Generative Adversarial Networks for learning and loading random distributions". In: *npj Quantum Information* 5.1 (Nov. 22, 2019), p. 103. ISSN: 2056-6387. DOI: 10.1038/s41534-019-0223-2. URL: https://www.nature.com/articles/s41534-019-0223-2 (visited on 11/21/2024).