Synthetic Asset Price Paths Generation Using Denoising Diffusion Probabilistic Model

by

Shujie Liu

A research paper
presented to the University of Waterloo
in fulfillment of the
research paper requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Waterloo, Ontario, Canada, 2023

Author's Declaration

Abstract

Synthetic asset price paths are crucial in various financial applications, such as risk management and strategy testing. In this study, we present the application of a Denoising Diffusion Probabilistic Model (DDPM) to generate synthetic asset price paths. The objective is not to exactly replicate the original paths, but rather to capture their underlying dynamics, thereby creating plausible yet unseen scenarios. Our methodology innovatively incorporates a discrete cosine transform, which allows the DDPM to learn in the frequency domain, substantially improving its learning efficacy. Unlike traditional approaches for synthesizing asset price paths involving predefined assumptions about the original paths' price dynamics, our approach avoids explicit model selection and calibration. Through both qualitative and quantitative assessments, we show that the synthetic paths generated by our DDPM closely align with the dynamics of the original paths, thereby affirming the effectiveness of our approach.

Acknowledgements

I am very grateful to my supervisor, Professor Justin Wan, for his invaluable guidance and insightful feedback throughout my research journey. I would also like to thank Professor Christopher Batty for his time and expertise in serving as the second reader of this paper.

My sincere thanks also go to my high school math teacher, Mr. Hazon Rami, who inspired me and believed in my pursuit of higher education in mathematics.

Dedication

I dedicate this work to my parents, my brother, and my partner, Sky Qiao, whose unconditional love and support have been my constant source of strength and inspiration.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The financial market, despite its general data abundance, often faces data scarcity challenges in many applications. This issue primarily arises from our reliance on historical data, which limits our understanding to scenarios that have already occurred, without access to other potential scenarios. For instance, while market crashes are a primary concern for investors, they are infrequently represented in historical data. Consequently, synthetic market price data can be useful in testing and validating the robustness of trading strategies in hypothetical market crashes [2]. Beyond strategy testing, synthetic asset price data is also valuable in areas such as risk management, portfolio construction, and regulatory compliance [15, 2, 27].

Traditional methods for creating synthetic asset prices typically involve selecting a model to describe price dynamics, calibrating its parameters, and then using it to generate new data. This approach, however, comes with considerable challenges. Designing a model that accurately reflect the complex nature of asset price movements is difficult. Additionally, the calibration process is prone to inaccuracies, especially when calibration data is limited.

The recent advancement of deep generative models in time series applications has opened new possibilities for producing synthetic asset prices. These models, based on neural networks, offer a 'model-free' approach as they do not rely on predefined assumptions about the dynamics of the studied time series. Notable successes have been achieved with generative adversarial networks (GANs) in time-series applications, as seen with TimeGAN [9, 37]. However, GAN-based methods can be difficult to train and may face stability and model collapse issues.

In this study, we explore the Denoising Diffusion Probabilistic Model (DDPM), which outperformed GANs in image synthesis [14]. Although generative diffusion models in general have seen applications in various time series tasks, their use in time series generation remains limited [36]. Here, we focus on applying DDPM to synthesize asset price data. We introduce a novel preprocessing step involving discrete cosine transform, which allows the DDPM to learn in the frequency domain and significantly enhances its ability to capture the dynamics of asset prices.

This paper is structured as follows: Chapter 2 presents necessary background information. Chapter 3 formulates the problem and explains our methodology. Chapter 4 describes the evaluation criteria used for the generated asset price paths. Chapter 5 discusses the results and analysis of these generated paths. Finally, Chapter 6 concludes the study, highlighting the main findings and contributions.

Chapter 2

Background

In this chapter, we will discuss the essential background knowledge related to our study. We begin with an introduction of deep generative models, focusing particularly on the DDPM. Subsequently, we turn our attention to two stochastic asset price models: the Geometric Brownian Motion and the Heston Stochastic Volatility Model. These models form the foundational theoretical framework for generating our synthetic training datasets. Furthermore, we will discuss methods for European option pricing, which is relevant for evaluating the efficacy of the DDPM in generating synthetic asset price paths.

2.1 Deep Generative Models

Deep generative models, a subset of deep neural networks, are designed to capture the characteristics of existing data and subsequently generate new data that bear similarities but are not identical to the original. This process typically involves approximating the training data distribution p with an estimated distribution $\tilde{p}$, based on observed x, and then sampling $\hat{x}$ from $\tilde{p}$ [5]. There exists a variety of generative models, including variational autoencoders [22], generative adversarial networks [10], and generative diffusion models [36]. These models are applied in a wide range of fields, for example, image and audio synthesis, natural language processing, and temporal data modeling [23, 17, 36]. In this section, our focus will be on the DDPM, a specific type of generative diffusion model [14].

2.1.1 Denoising Diffusion Probabilistic Model

The introduction of the DDPM [14] led to an advancement in image synthesis, enhancing image fidelity and surpassing the performance of GANs. Following this innovation, a variety of text-to-image generation models have been developed, building upon the DDPM framework. Notable examples include Stable Diffusion and Midjourney [26, 29].

At its core, DDPM operates by incrementally introducing noise into a set of original data during a forward process and subsequently learning to reverse this noise addition in a backward process. This enables the generation of new data from Gaussian noise, that aligns with the same probability distribution as the original data. We summarize this process into three phases: 1) the forward process, 2) the reverse process, and 3) the sampling process. In the following subsections, we will provide a detailed exploration of each of these phases.

Forward Process

In the forward process of DDPM, we assume the observed data points $\mathbf{x}_0$ are sampled from a probability distribution q. This process employs a Markov chain [6] to incrementally add Gaussian noise $\mathcal{N}(\mathbf{0}, \mathbf{I})$ to $\mathbf{x}_0$, following a variance schedule $\beta_1, ..., \beta_T$, to generate a sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$. For timesteps $t = 1, ..., T$, the transition probability is defined as:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) \sim \mathcal{N}\left(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right).$$

Hence, given $\mathbf{x}_{t-1}$, $\mathbf{x}_t$ can be obtained by:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_t,$$

where $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. As the final timestep T becomes large, the last data point $\mathbf{x}_T$ converges to $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

The use of Gaussian noise simplifies the derivation of the joint probability distribution $q(\mathbf{x}_{1,\ldots,T}|\mathbf{x}_0)$ for the entire forward process as:

$$q(\mathbf{x}_{1,\ldots,T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{t=1}^{T} \mathcal{N}\left(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right),$$

which implies

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}}\mathbf{x}_0, (1-\bar{\alpha})\mathbf{I}\right), \tag{2.1}$$

where

$$\alpha_t = 1 - \beta_t, \bar{\alpha} = \prod_{s=1}^{t} \alpha_s.$$

This closed-form representation enables direct sampling of $\mathbf{x}_t$ for any timestep t from $\mathbf{x}_0$ when given a variance schedule $\beta_1, ..., \beta_T$.

The original work on DDPM [14] employed a linear variance schedule. However, subsequent research by [25] demonstrated that such a linear schedule tends to destroy information more rapidly than necessary during the forward process. Thus, we utilized a cosine variance schedule as in [25]. This modification aims to preserve information more effectively throughout the forward process, potentially enhancing the model's performance in generating new data.

Reverse Process

The reverse process attempts to counteract the noise addition at each timestep of the Markov chain. It is modeled by a neural network parameterized by θ, which takes noisy inputs $\mathbf{x}_t$ and timestep t, and learns to output approximated Gaussian distributions

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \sim \mathcal{N}(\mu_\theta(\mathbf{x}_t, t), \mathbf{\Sigma}_\theta(\mathbf{x}_t, t)),$$

effectively reversing the noise added in the transition from $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$. For a well-trained network, the reconstructed distribution $p_\theta(\mathbf{x}_0)$ from the reverse process should match the original data distribution $q(\mathbf{x}_0)$. By initializing noisy inputs $\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we can generate synthetic samples $\hat{\mathbf{x}}_0$ from Gaussian noise.

To approximate the parameterization for p_θ, we need to estimate the mean μ_θ and the covariance $\mathbf{\Sigma}_\theta$. The original DDPM paper [14] advises setting $\mathbf{\Sigma}_\theta(\mathbf{x}_t, t)$ to time-dependent constants $\sigma_t^2\mathbf{I}$ based on the variance schedule $\beta_1, ..., \beta_T$, where

$$\sigma_t^2 = \tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\beta_t.$$

The remaining task is to set $\mu_\theta(\mathbf{x}_t, t)$ appropriately.

Given $\mathbf{x}_0$, we can reverse the forward process to retrieve the 'true' probability distribution, represented as follows:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I}),$$

where

$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t.$$

Ideally, we aim for $\mu_\theta = \tilde{\mu}_t$. However, since $\mathbf{x}_0$ is not available during the reverse process, we must estimate it. In the literature, there are three common approaches:

1. Predict initial state: Predict $\mathbf{x}_0$ directly [14].
2. Predict noise: Using (2.1), we can represent $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Then, given the noise ϵ, we can recover $\mathbf{x}_0$. Hence, we can predict ϵ with ϵ_θ instead of predicting $\mathbf{x}_0$ [14].
3. Predict v-parameterization: Following the suggestion by [31], we can predict the $\mathbf{v}$-parameterization, where $\mathbf{v} = \alpha_t\epsilon - \sigma_t\mathbf{x}_0$, leading to more stable sampling outcomes and a faster sampling process.

Sampling Process

Upon the successful training of the reverse process, the sampling procedure is straightforward. We start with a sample drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I})$, representing the noisy data $\hat{\mathbf{x}}_T$ at the final timestep. Then, given $\hat{\mathbf{x}}_t$, we can obtain $\hat{\mathbf{x}}_{t-1}$ by applying the learned reverse process as follows:

$$\hat{\mathbf{x}}_{t-1} = \mu_\theta(\hat{\mathbf{x}}_t, t)\hat{\mathbf{x}}_t + \sigma_t\hat{\mathbf{z}}_t,$$

where $\hat{\mathbf{z}}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This process sequentially 'denoises' the data by reversing the Markov chain from $t = T$ back to $t = 0$. The final denoised output $\hat{\mathbf{x}}_0$ is the synthetic data generated by the DDPM.

2.2 Asset Price Models

Predicting the movements of financial asset prices is a challenging problem. Empirical studies indicate that financial asset prices exhibit significant randomness [3]. Consequently, stochastic processes are often employed to model the behavior of financial asset prices [28, 7]. In this section, we introduce two stochastic asset price models based on stochastic differential equations (SDEs): the Geometric Brownian Motion (GBM) and the Heston Stochastic Volatility model (Heston model).

2.2.1 Geometric Brownian Motion

We begin our discussion with a definition of the GBM:

Definition: A GBM S_t is a continuous-time stochastic process that satisfies the following SDE:

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t,$$

where μ is called the drift rate, σ is called the volatility, and W_t is a Wiener process [24]. Here, we assume μ and σ are constants. Using Ito's lemma [16], this SDE has a unique solution:

$$S_t = S_0 \exp\left((\mu - \frac{\sigma^2}{2})t + \sigma W_t \right).$$

This closed-form solution can be discretized to get the following update rule for S_{t+1} given S_t, where Δt represents a time increment, and $Z_t \sim \mathcal{N}(0,1)$:

$$S_{t+1} = S_t \exp\left(\left(\mu - \frac{1}{2}\sigma^2 \right) \Delta t + \sigma \sqrt{\Delta t} Z_t \right) \tag{2.2}$$

With this update rule, we can simulate an asset price path of length N given (S_0, μ, σ, T) using Algorithm 1 (see Figure 2.1 for an illustration). This will enable us to construct a training set representing GBM dynamics in Chapter 3.

Algorithm 1 Simulation of an asset price path following GBM dynamics

```
   Input: $S_0, \mu, \sigma, N, T$
   Output: An asset price path of length $N$ following GBM dynamics
1: Initialize an array $S$ with $S[0] = S_0$
2: $\Delta t \leftarrow \frac{T}{N}$                                   ▷ Time increment
3: for $i = 1$ to $N - 1$ do
4:     $Z \leftarrow$ Sample from $\mathcal{N}(0,1)$
5:     $S[i] \leftarrow S[i-1] \exp\left( (\mu - \frac{\sigma^2}{2})\Delta t + \sigma\sqrt{\Delta t} Z \right)$
6: end for
7: return $S$
```

GBM is widely applied to model asset price dynamics and is used in the Black-Scholes model [4]. However, GBM does not perfectly capture real-world asset price dynamics, as it is based on several simplifying assumptions. GBM assumes that the asset log returns follow a normal distribution, while empirical studies found that returns are often skewed and have fatter tails [35]. In addition, from empirical asset price paths, we observe behaviors such

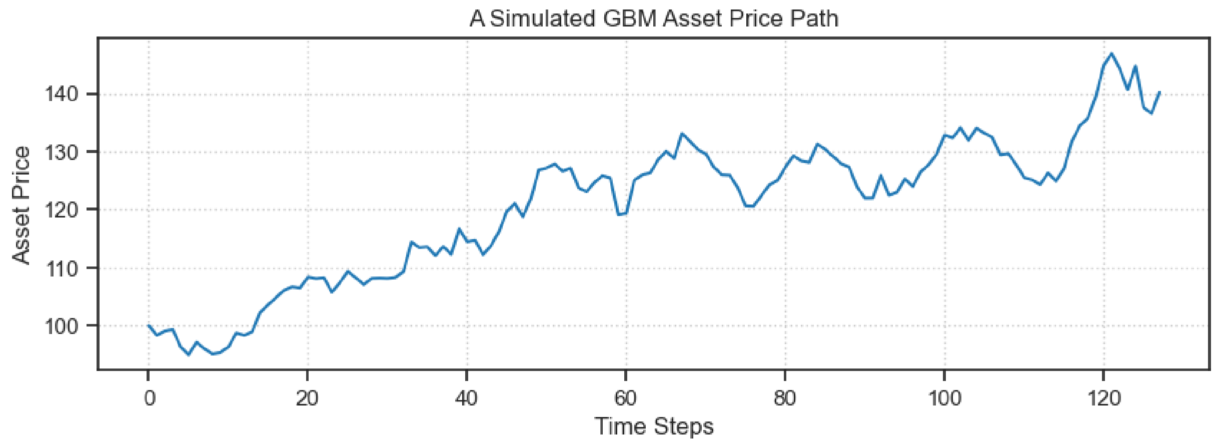

Figure 2.1: A simulated asset price path using GBM with parameters $S_0 = 100$, $\mu = 0.05$, $\sigma = 0.3$, $N = 128$, $T = 0.5$.

as varying volatility levels through time, sudden jumps [19], and regime switching [12]. Various models are proposed to better capture these dynamics, and in the next section, we will discuss one of such models – the Heston model – which captures the varying volatility level behavior by modeling volatility as a stochastic process.

2.2.2 Heston Stochastic Volatility Model

We start with a definition of the Heston model:

Definition: A continuous-time stochastic process S_t is said to follow the Heston model if it satisfies the following SDE:

$$dS_t = \mu S_t\, dt + \sqrt{v_t} S_t\, dW_t^{(1)}, \tag{2.3}$$

where μ is a constant representing the drift rate for S_t, and v_t is a stochastic process representing the variance at time t satisfying the following SDE:

$$dv_t = \kappa(\theta - v_t)\, dt + \xi\sqrt{v_t}\, dW_t^{(2)},$$

where θ represents the long-term variance, κ represents the variance mean reversion rate, and ξ represents the volatility of the volatility (note that volatility is calculated as $\sqrt{v_t}$). In these two SDEs, $dW_t^{(1)}$ and $dW_t^{(2)}$ are Wiener processes with correlation ρ, where $dW_t^{(1)} dW_t^{(2)} = \rho\, dt$.

Applying Ito's lemma [16], the exact solution to the SDE (2.3) can be obtained by:

$$S_t = S_0 \exp\left(\int_0^t \left(\mu - \frac{1}{2}v_s\right) ds + \int_0^t \sqrt{v_s}\, dW_s^{(1)}\right).$$

Using the Euler discretization scheme [34], the exact solution can be discretized to obtain the following update rules:

$$S_{t+1} = S_t \exp\left(\left(\mu - \frac{v_t}{2}\right)\Delta t + \sqrt{v_t \Delta t} Z_t^{(1)}\right) \tag{2.4}$$

$$v_{t+1} = v_t + \kappa(\theta - v_t)\Delta t + \xi\sqrt{v_t \Delta t} Z_t^{(2)} \tag{2.5}$$

With these update rules, when we are given $(S_0, v_0, T, \mu, \kappa, \theta, \xi, \rho)$, where S_0 represents the initial price and v_0 represents the initial variance, we can simulate an asset price path and a volatility path each of length N using Algorithm 2 (see Figure 2.2 for an illustration). This will help us construct a training set in Chapter 3 representing the Heston dynamics.

Algorithm 2 Simulation of an asset price path following Heston dynamics

```
    Input: $\{S_0, v_0, \mu, \kappa, \theta, \xi, \rho, N, T\}$
    Output: An asset price path of length N following Heston dynamics
 1: Initialize arrays $S$ with $S[0] = S_0$ and $v$ with $v[0] = v_0$
 2: $\Delta t \leftarrow \frac{T}{N}$                                         ▷ Time increment
 3: Set $\mu_{\text{vec}} = [0, 0]$ and $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$      ▷ Parameters for correlated Brownian motions
 4: for $i = 1$ to $N - 1$ do
 5:     Sample $Z$ from $\mathcal{N}(\mu_{\text{vec}}, \Sigma)$          ▷ Correlated standard normal random variables
 6:     $S[i] \leftarrow S[i-1] \exp\left((\mu - \frac{v[i-1]}{2})\Delta t + \sqrt{v[i-1]\Delta t} Z[0]\right)$
 7:     $v[i] \leftarrow \max(v[i-1] + \kappa(\theta - v[i-1])\Delta t + \xi\sqrt{v[i-1]\Delta t} Z[1], 0)$   ▷ variance cannot
    be negative
 8: end for
 9: return $S$
```

Despite the greater flexibility the Heston model allows for in modeling volatility compared to the GBM, it still operates under certain assumptions that may not align with real market behaviors. For example, the Heston model assumes that volatility is mean-reverting at a constant rate. This assumption can lead to the model's inability to accurately produce extreme volatility paths during market crises, which are often observed in real financial markets [20].

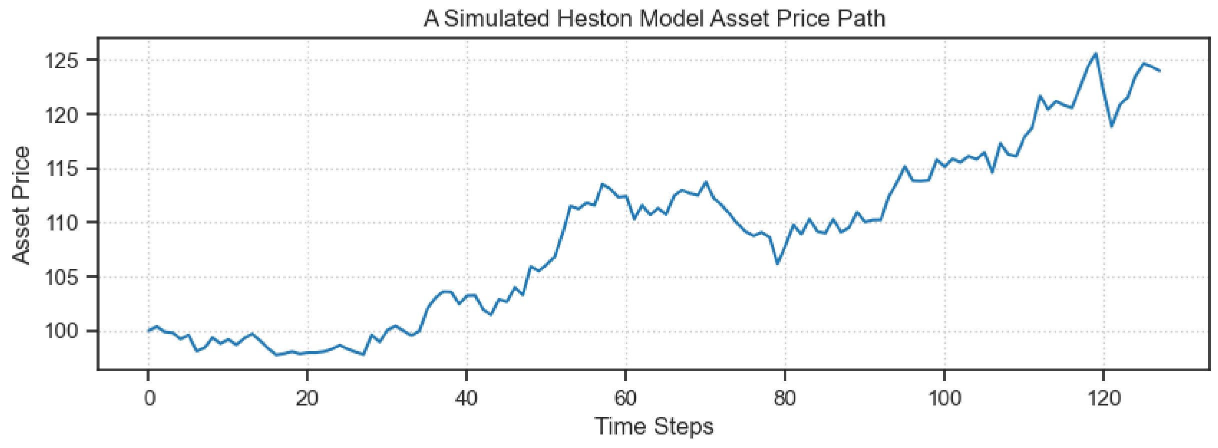

Figure 2.2: A simulated asset price path using Heston model with parameters $S_0 = 100$, $v_0 = 0.1^2$, $\mu = 0.05$, $\kappa = 1$, $\theta = 0.3^2$, $\xi = 0.1$, $\rho = 0.5$, $N = 128$, $T = 0.5$.

2.3 European Option Pricing Problem

A European option is a financial contract that grants the holder the right, but not the obligation, to trade an underlying asset at a predetermined price (strike price K) in a predetermined time (time to maturity T). There are two types of European options: a call option, granting the right to purchase, and a put option, granting the right to sell. In this study, we focus on European call options. On the maturity date, if the asset price S_T exceeds the strike price K, the call option holder can exercise the option, buying the asset at K and selling it in the open market at S_T, realizing a profit of $\max(S_T - K, 0)$. Conversely, if $S_T \leq K$, exercising the option is not economically beneficial. The call option payoff is therefore $\max(S_T - K, 0)$. A call option is termed in-the-money (ITM) when the strike price K is less than the current market price S_0; at-the-money (ATM) if $K = S_0$; and out-of-the-money (OTM) if $K > S_0$.

When the asset price follows a GBM dynamic, the Black-Scholes formula [4] can be used to obtain the price of a European call option as:

$$C(S_t, t) = S_t N(d_1) - K e^{-r(T-t)} N(d_2), \tag{2.6}$$

where

- C is the price of the call option,

- S_t is the price of the underlying asset at time t,
- K is the strike price of the option,
- T is the time to maturity of the option,
- t is the current time,
- r is the risk-free interest rate,
- $N(\cdot)$ is the cumulative distribution function of the standard normal distribution,
- d_1 and d_2 are given by

$$d_1 = \frac{\ln\left(\frac{S_t}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T - t)}{\sigma\sqrt{T - t}}, \tag{2.7}$$

$$d_2 = d_1 - \sigma\sqrt{T - t}, \tag{2.8}$$

- and σ is the volatility of the underlying asset's returns.

Note that the Black-Scholes formula is derived to ensure the option price obtained is arbitrage-free, and thus we refer to it as the theoretical 'no-arbitrage' price [4].

If the underlying asset price follows the Heston model, an analytic solution for the no-arbitrage option price is not available [13]. However, the Monte Carlo method provides a numerical approach for estimation [21]. This method involves simulating a large number of asset price paths using discretized Heston model update rules (2.4) and (2.5) and computing the expected value of the discounted payoffs from these paths. Specifically, when these paths are simulated under the assumption that their drift rate equals the risk-free rate r, discounting the payoffs at the same rate r and averaging them yields a value identical to the no-arbitrage price as the number of paths M approaches infinity [18, 21]. In practice, we set M to be a large number (e.g., $M = 1,000,000$) to get a close estimation of the no-arbitrage price.

This Monte Carlo approach of approximating the no-arbitrage price is also applicable when the asset prices evolve with a GBM dynamic, in which case the asset price paths can be simulated using discretized GBM update rules such as (2.2).

It is important to note that in actual financial markets, the drift rates of asset prices typically do not align with the risk-free rate r. Therefore, we think of the asset price paths simulation under the assumption of a drift rate equal to r as if we were in a 'risk-neutral' world, where all assets are presumed to grow at rate r. While this 'risk-neutral' world is hypothetical and not realistic, it is useful for obtaining the no-arbitrage option price using the simplistic Monte Carlo method.

Chapter 3

Methodology

In this chapter, we will formulate our problem and discuss the process for generating synthetic asset price paths using the DDPM. We begin by compiling a large training set of asset price paths, which is either generated synthetically or resampled from market prices. We then discuss our approach to preprocess these data, including a novel application of the discrete cosine transform, to facilitate DDPM's learning. Next, we detail the training and sampling process. We will conclude with a description of the post-processing steps, which convert the generated data into synthetic asset price paths.

Throughout this chapter, we refer to the asset price paths in the training sets as the 'original asset price paths', and refer to the ones generated from the learned DDPM as the 'generated asset price paths'.

3.1 Problem Formulation

Let S_t represent the price of an asset at time t, and consider an asset price path $\{S_t\}_{t=0}^{N-1}$ with length N and initial price S_0. Suppose this asset price is governed by an underlying stochastic process. When two asset price paths $\{S_t^i\}_{t=0}^{N-1}$ and $\{S_t^j\}_{t=0}^{N-1}$ are governed by the same underlying stochastic process, we say they share the same price dynamic. With this setup, we are ready to formulate our problem:

Problem 1 (Synthetic asset price path generation problem)**.** *Given an asset price path $\{S_t\}_{t=0}^{N-1}$, how to generate a synthetic asset price path $\{\hat{S}_t\}_{t=0}^{N-1}$ that replicates the original path's underlying stochastic dynamics?*

The core challenge in this problem arises from the fact that the underlying stochastic process of $\{S_t\}_{t=0}^{N-1}$ is not directly observable and must be inferred from empirical data. Traditionally, this challenge is addressed by approximating the process with SDEs, calibrating these equations with empirical data, and then synthesizing a path using the calibrated models. However, choosing suitable SDEs for modeling asset price dynamics is a complex task. As discussed in Chapter 2, both GBM and the Heston model have inherent limitations and do not perfectly capture real asset price dynamics. In addition, the model parameter calibration process is prone to errors, especially when empirical data for calibration is limited.

In contrast to the traditional approach, we propose to utilize a DDPM and leverage the capability of neural networks to learn complex mappings. By training the DDPM to learn asset price dynamics, we can directly generate synthetic asset price paths from the trained model, eliminating the need for explicit model selection or calibration. It is worth noting that our goal is not merely creating paths identical to the original one, but to capture the dynamics of the original path's price evolution, allowing for the generation of new, yet plausible, price paths that could have been observed under the same dynamic conditions.

However, training a DDPM effectively requires a substantial training set instead of just one original asset price path. Therefore, in the following section, we explore the methodologies for constructing such training sets.

3.2 Training Set Construction

In this section, our objective is to assemble a training set of M asset price paths, each of length N, all governed by the same stochastic process. Ideally, M should be sufficiently large to ensure effective training (e.g., $M = 10,000$). This task is straightforward if we consider training sets comprised of synthetic data, as we can simulate numerous paths using a chosen stochastic process and a fixed set of defining parameters. However, if we want to construct training sets containing real market data, techniques like bootstrapping are necessary. Once we have the M paths, we then preprocess them to enhance the training for DDPM.

3.2.1 Synthetic Data

For synthetic data, we consider simulating asset price paths using a chosen stochastic process introduced in Chapter 2, i.e., the GBM or the Heston model. Recall that given parameters $\{S_0, \mu, \sigma, N, T\}$, the GBM uses the discretized update rule (2.2) to evolve from S_t to S_{t+1}; given parameters $\{S_0, v_0, \mu, \kappa, \theta, \xi, \rho, N, T\}$, the Heston model uses the discretized update rules (2.4) and (2.5) to evolve from S_t to S_{t+1}. With these update rules, for each stochastic process and each set of defining parameters, we simulate M paths as our training set. Each path, represented as $\{S_t^i\}_{t=0}^{N-1}$ for $i = 1, \ldots, M$, is a discrete sequence of length N, where S_t^i denotes the simulated price at time t for the i^{th} path. These M paths share the same specified dynamic, but each path has its unique realized values due to the inherent randomness in both the GBM and Heston models.

3.2.2 Real Market Data

For real market data, we face the challenge of only having access to a single realized path for an asset whose underlying dynamic is unknown. Unlike synthetic data, we cannot directly observe a set of potential paths that the asset price might have followed. To overcome this limitation, we employ a bootstrapping technique to generate a representative sample of paths from the single observed path [11]. We start by gathering a long historical price path for an asset from Yahoo Finance[1], denoted by $\{S_t\}_{t=0}^{N_{\text{long}}-1}$, where N_{long} is the length of this path. We operate under the assumption that the dynamic of this path remains constant over time, i.e. the segment $\{S_t\}_{t=t_1}^{t_j}$ has the same dynamic as the segment $\{S_t\}_{t=t_1+k}^{t_j+k}$ for $t_1 \leq t_j = 0, \ldots, N-1$ and $k = 0, \ldots, N-1-t_j$, thereby enabling the resampling of M paths of length N (with $N \leq N_{\text{long}}$) that mirror the original path's dynamics. This resampling is detailed in Algorithm 3.

3.2.3 Preprocessing With Discrete Cosine Transform & Mirror Reflection Extension

Discrete Cosine Transform

If we train DDPM on asset price paths without any data preprocessing, it can be challenging for DDPM to effectively capture the underlying dynamics. Figure 3.1 illustrates a comparison of five original paths against five generated paths from a fine-tuned DDPM trained on raw asset price paths. The generated paths exhibit unnatural boundary spikes and fail to mimic the original volatility dynamics. This issue arises because within each training set, despite sharing the same dynamic, individual paths vary significantly in values

[1]Data retrieved from Yahoo Finance (https://finance.yahoo.com/)

Algorithm 3 Training Set Construction With Bootstrapping

```
    **Input:** Historical price path $\{S_t\}_{t=0}^{N_{\text{long}}-1}$, number of desired short paths $M$, short path size $N$ with $N \leq N_{\text{long}}$
    **Output:** $M$ bootstrap paths of size $N$, each denoted by $\{S_t'^m\}_{t=0}^{N-1}$, for $m = 1, \ldots, M$
1:  Initialize an empty list, Paths, to store bootstrapped paths.
2:  for $m = 1$ to $M$ do
3:      Compute return series $R$ from the historical prices $\{S_t\}_{t=0}^{N_{\text{long}}-1}$.
4:      Create an empty list, BootstrapReturns
5:      while len(BootstrapReturns) $\leq N - 1$ do
6:          Sample a block size $k$ from a geometric distribution.
7:          Let $k = \min(k, N - 1 -$ len(BootstrapReturns))
8:          Select a contiguous block of returns of size $k$ from $R$ at random.
9:          Append the selected block to BootstrapReturns.
10:     end while
11:     Reconstruct the price path $\{S_t'^m\}_{t=0}^{N-1}$ from BootstrapReturns and add to Paths.
12: end for
```

at each timestep due to the randomness of asset prices. As a result, the DDPM struggles to determine the appropriate values for each path, leading to the generation of paths where all paths exhibit similar value ranges at each timestep, as shown in Figure 3.1.

Similar challenges are also encountered in fields such as signal analysis, where signals can be complex and noisy, exhibiting a wide range of variation in values. To tackle such issues, signals are often processed using tools like the Discrete Fourier Transform (DFT) [33, 8], which simplifies the analysis by decomposing signals into constituent frequencies. Inspired by this, we adopt a similar approach by applying the discrete cosine transform (DCT) [1] to preprocess the raw asset price data. DCT is a variant of the DFT, but using only real numbers, making it more suitable for our purpose. Given a sequence $\{f_n \in \mathbb{R} | n = 0, \ldots, N-1\}$, DCT converts it into a sequence of cosine coefficients of the same length $\{C_k \in \mathbb{R} | k = 0, \ldots, N-1\}$ by:

$$C_k = \sum_{n=0}^{N-1} f_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right], \quad k = 0, 1, \ldots, N-1.$$

We refer to the original sequence f_n as the time-domain representation of the data, and refer to the cosine coefficients C_k as the frequency-domain representation. DCT is equipped with an inverse operation, namely the inverse discrete cosine transform (IDCT), which allows reconstruction of the original sequence from the cosine coefficients.

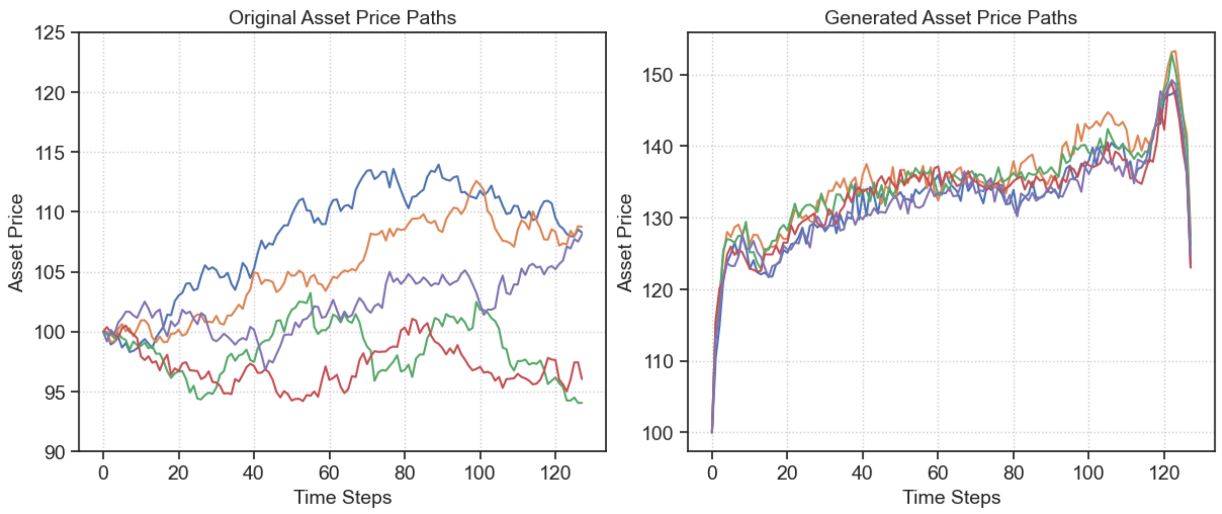

Figure 3.1: Original paths vs. generated paths, where DCT is not applied to the training set of DPPM. Original paths are simulated using GBM with parameters $S_0 = 100, \mu = 0.05, \sigma = 0.1, N = 128, T = 0.5$.

Transforming the raw asset price paths into the frequency domain through DCT allows us to analyze the frequency patterns that are generally more stable than the raw time-domain data. In Figure 3.2, we illustrate this by comparing the time-domain representation and the frequency-domain representation of two price paths governed by the same GBM price dynamics. The original time-domain paths appear very different, exhibiting significant variability across all timesteps, yet their frequency-domain representations reveal more consistent patterns. Specifically, the low-frequency components representing the core patterns of the data have higher DCT coefficients, while the high-frequency components which often correspond to noise have values close to 0. This highlights the effectiveness of frequency domain analysis in uncovering fundamental similarities, which can simplify the learning for DDPM.

Applying DCT to each path in our training set, we obtain a set of transformed paths of length N. Figure 3.3 compares five original asset price paths with five DDPM generated paths, where the DCT is applied to all the paths in the training set of the DDPM. We observe significant improvement in generated paths' alignment with the original paths' dynamics. However, there are minor discrepancies at the left boundaries of the generated paths, characterized by sudden spikes, which are not present in the original paths.

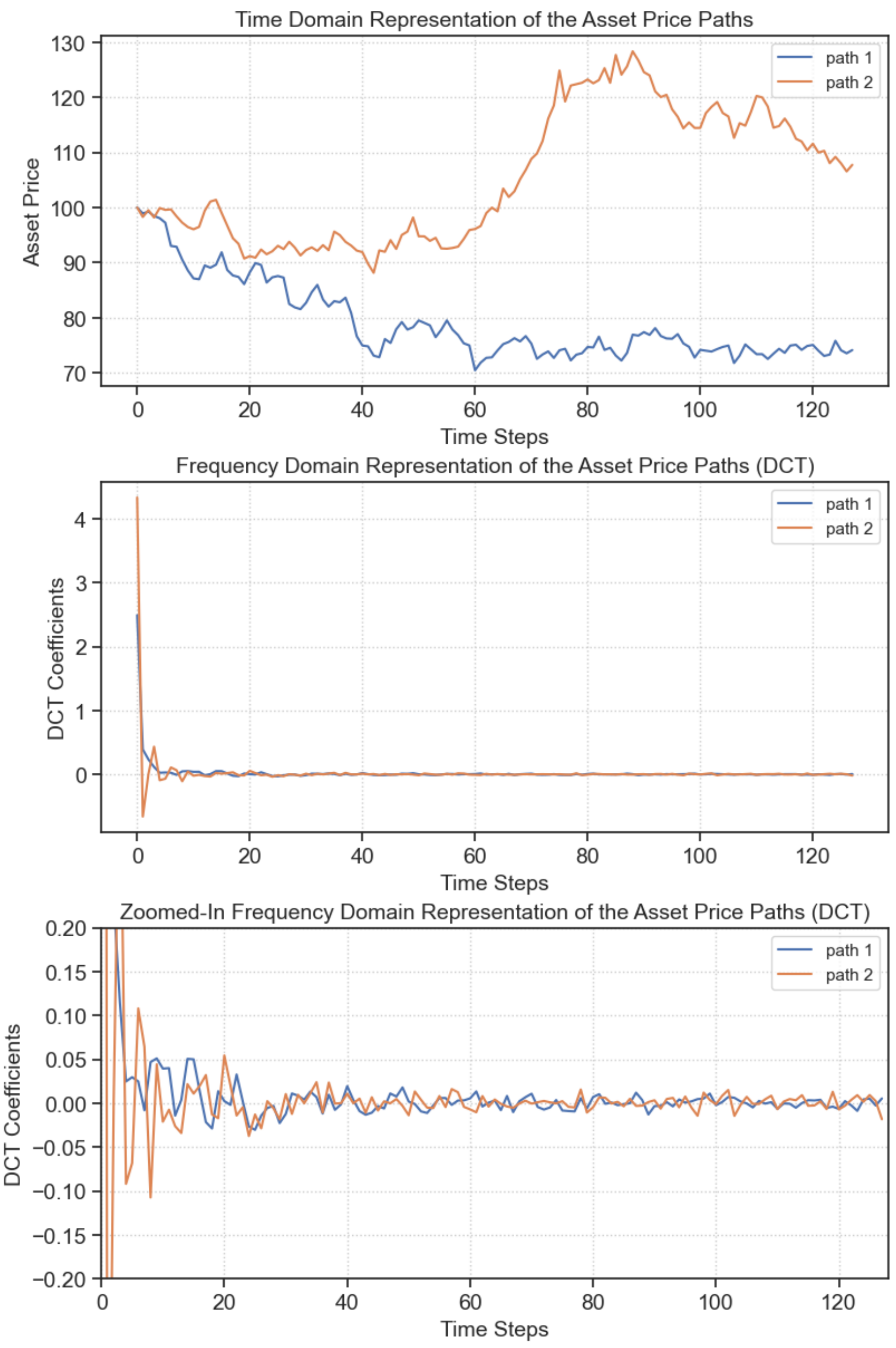

Figure 3.2: Time-domain vs. frequency-domain representation of asset price paths. Time-domain paths are simulated using GBM with parameters $S_0 = 100, \mu = 0.05, \sigma = 0.1, N = 128, T = 0.5$.

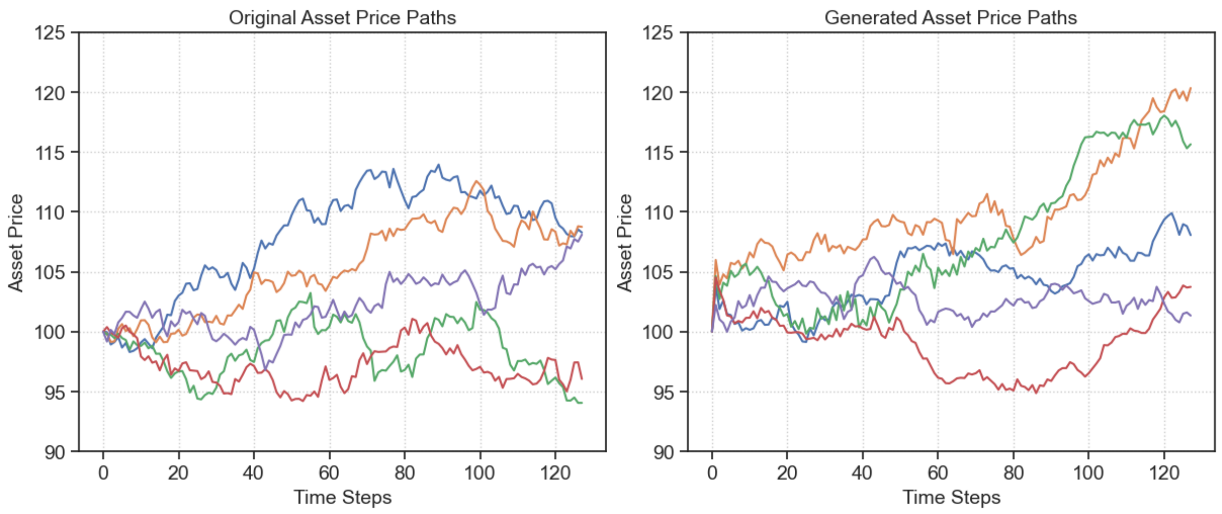

Figure 3.3: Original paths vs. generated paths, where DCT is applied to the training set of DPPM. Original paths are simulated using GBM with parameters $S_0 = 100, \mu = 0.05, \sigma = 0.1, N = 128, T = 0.5$.

Mirror Reflection Extension

Typically, DCT are applied to input signals presumed to be periodic over a period N with the condition $f_0 = f_{N-1}$. However, the time series representing asset price paths in our study are not inherently periodic. Simply extending each path with a value equal to S_0 could result in an abrupt transition at the last point of the original data set, which could misrepresent the signal's natural progression. To address this, we utilize mirror reflection, appending the time-reversed paths to the end of the original. This technique ensures the extended paths do not introduce abrupt transition, while achieving the desired periodicity with a period length of $2N$ and the condition $S_0 = S_{2N-1}$, thereby making them suitable for DCT analysis. As we will see in Section 3.4 and Figure 3.4, this mirror reflection extension can help us address the boundary issues observed previously in Figure 3.3.

Using mirror reflection extension in conjunction with the DCT, we construct a training set comprising M paths represented in the frequency domain, each extended to a length of $2N$. With this pre-processing step completed, we proceed to the training phase.

3.3 Training

Training a DDPM entails a forward process where noise is systematically introduced and a reverse process where a neural network learns to revert this noise addition. For this reverse process, we employ the same model architecture as in the original DDPM study, namely a U-Net architecture [14]. While U-Net was initially developed for medical image segmentation [30], its superior performance in capturing both local and global information makes it well-suited for the denoising tasks in DDPM. The U-Net processes inputs at various noise levels and their corresponding timesteps, and it learns to predict the noise added at each forward step. Considering the original DDPM work focused on 2D image generation and our specific task involving 1D time series data, we adopted a 1D U-Net architecture where the number of input channel is set to 1 to accommodate the dimensional differences.

Other model architectural configurations and hyperparameter choices for the DDPM are determined by a thorough grid search, detailed in Table 3.1. We set the length of each asset path in the training set to be $N = 2^7 = 128$. While N can be any positive integer, we select $N = 128$ to align with the requirement of the U-Net architecture, which prefers step lengths to be the power of 2.

Configurations	**Choices**	**Optimal Choice**
Training Set Size	10^4, 10^5	10^4
Batch Size	10, 50, 100, 200	100
Learning Rate	10^{-4}, 10^{-5}, 10^{-6}	10^{-5}
Number of Epochs	50, 100, 200	100
Sampling Timesteps	1000, 2000, 3000	3000
Training Objective [1]	Predict noise; predict initial state; predict v-parameterization	predict v-parameterization

[1] As discussed in Section 2.1.1.

Table 3.1: Grid search configuration choices and their identified optimal choices for the DDPM.

3.4 Sampling & Post-processing

Upon completion of the training, we can generate synthetic signals from the trained DDPM, which we refer to as the 'generated data'. Since we performed DCT on the original asset price paths in the training set, the generated data mimics the dynamics of original paths' frequency-domain representations. Consequently, to reconstruct the time-domain representation of the data, we apply IDCT to the generated data.

Since the asset price paths in the training data are extended with mirror reflection as shown by the five exemplary paths in the first plot of Figure 3.4, the IDCT of the generated data should also reflect symmetry. The second plot in Figure 3.4 shows the time-domain representation of the generated data, which is almost symmetrical, demonstrating DDPM's effectiveness in capturing symmetry. We observe that the 'first half' of the generated data (from $t = 0$ to $t = N-1$) exhibits atypical boundary behavior. In contrast, the 'second half' (from $t = N$ to $t = 2N - 1$) which corresponds to the portion learned from the mirrored original, more closely aligns with the original data's dynamics after flipping, particularly at the boundaries. Hence, we select the 'second half' as the more accurate generated paths.

It is also worth noting that, all paths in the training set commence from the same initial value S_0. The sampled paths approximate this starting value, with minor deviations resulting from denoising inaccuracies. Such deviations are less critical since our primary focus is to model the price evolution. To reconcile this, we can adjust the paths to commence from our specified starting point S_0, ensuring consistency across the dataset. These adjusted paths are considered as our final generated asset price paths.

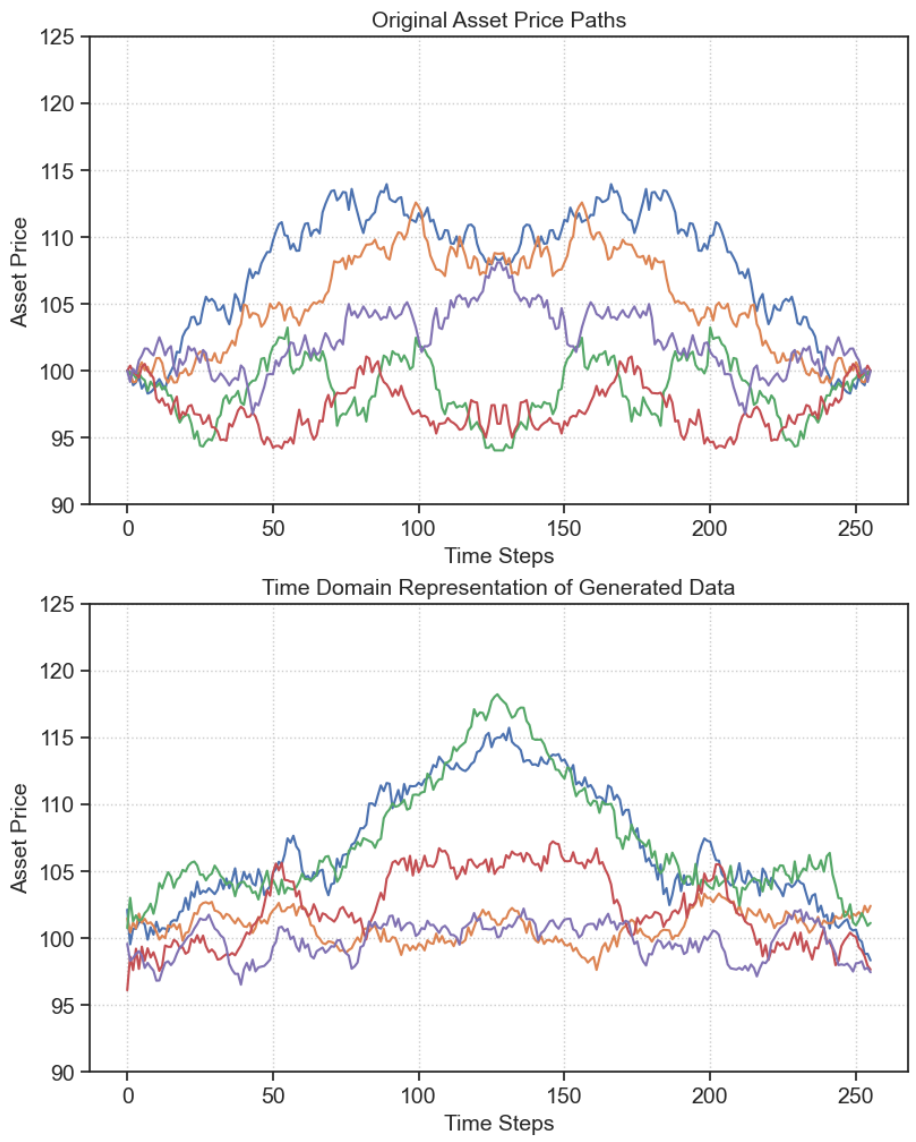

Figure 3.4: Original paths extended with mirror reflection vs. IDCT of the sampled data, which brings the sampled data from the frequency domain back to the time domain. Original paths are simulated using GBM with parameters $S_0 = 100, \mu = 0.05, \sigma = 0.1, N = 128, T = 0.5$.

3.5 Summary of Steps For Synthetic Asset Price Paths Generation

In this section, we provide a concise summary of the steps involved in our proposed methodology. These steps guide the DDPM to learn and subsequently generate new asset price paths that accurately reflect the dynamics of the original asset price series.

1. Begin with simulating or bootstrapping a set of asset price paths. Extend each path with its mirror reflection to create a periodic structure suitable for DCT.
2. Apply the DCT to the extended signals, thereby obtaining the cosine transform coefficients that encapsulate the frequency domain characteristics of the paths.
3. Train a DDPM using the model architectural configurations and hyperparameter choices listed in Table 3.1 to learn the characteristics of these cosine coefficients, which inherently learns to approximate the underlying stochastic process governing the original asset price paths.
4. Sample synthetic sets of cosine coefficients from the trained DDPM, which are analogous to the original data in the frequency domain.
5. Convert the sampled coefficients back to the time domain by applying the IDCT, resulting in the sampled paths.
6. Divide the sampled paths into two halves. Select and flip the 'second half' of each path, derived from the mirrored section of the extended path, as it generally yields a closer approximation to the original series.
7. Adjust the selected 'second half' of each path to start from the same initial value S_0, ensuring uniformity in the starting conditions across the generated synthetic dataset. These adjusted paths are considered our final generated synthetic asset price paths.

Chapter 4

Evaluation Criteria

Determining the extent to which the generated asset paths reflect the dynamics of the original asset paths requires careful assessment. Our aim is not to prove identicality between the series but to confirm they exhibit analogous dynamic patterns. We employ a combination of qualitative and quantitative methods for this purpose, which provides a robust framework for assessing whether the generated asset paths accurately capture the essence of the original paths' dynamics.

4.1 Qualitative Analysis

Similar to how synthetic images generated by DDPMs are assessed, where there is no exact 'original' for comparison, the realism of the generated images is often judged visually. For instance, common challenges such as accurately rendering images of hands in generative models are typically identified through visual inspection rather than quantitative analysis.

In our context, qualitative assessment involves examining the boundary behavior and overarching patterns of the generated asset price paths to gauge their resemblance to the original paths. For example, the generated paths presented in Figure 3.1 show noticeable issues at the boundaries and discrepancies in dynamics, thus they fail to capture the original paths' dynamics.

4.2 Quantitative Analysis

Our quantitative evaluation prioritizes the behavior of returns over direct analysis of asset price paths. This is because returns provide a normalized measure that facilitates comparison across different assets and time periods. In particular, log returns are particularly suitable for our analysis, as we assume continuous compounding for asset prices. Given asset prices S_t and S_{t+1}, the log return R_t is calculated as:

$$R_t = \log\left(\frac{S_{t+1}}{S_t}\right).$$

Kolmogorov-Smirnov Statistic

To determine the statistical resemblance between the log returns of the generated paths and the original paths, we employ the Kolmogorov-Smirnov (KS) test. This test assesses the null hypothesis that the two samples are drawn from the same distribution. The KS test provides two critical values: the KS-statistic, which quantifies the maximum discrepancy between the cumulative distribution functions of the two samples, and the KS p-value, which evaluates the significance of any observed differences.

The KS test is sensitive to large sample sizes, often rejecting the null hypothesis on small deviations. Hence, in our analysis, the KS-statistic is given precedence since our primary concern is identifying any substantial discrepancies. Minor differences, even if statistically significant, may not be as critical for the purpose of practical applications.

Drift and Volatility Analysis

We proceed to examine the drift and volatility inferred from the generated paths' log returns. Drift and volatility are the annualized mean and standard deviation, obtained by scaling the mean by $\frac{1}{dt}$ and standard deviation by $\frac{1}{\sqrt{dt}}$. We plot the distribution of the drift and volatility across all generated and original paths for a visual comparison of their distributional characteristics.

Autocorrelation Analysis

Autocorrelation is a fundamental aspect of time series analysis, reflecting the degree of correlation between the values of the series and their lagged selves. Autocorrelation is particularly useful in revealing repeating patterns or periodic signals within the data, with values near zero indicating a lack of such autocorrelation.

By setting a specific lag for the autocorrelation function and examining its value distribution across generated paths, we can examine the similarity in autocorrelation structure to the original paths. A close alignment in the distribution suggests that the time series share similar autocorrelation properties.

European Call Option Pricing Analysis

When original paths are simulated using GBM or the Heston model, an additional evaluation method can be applied to the generated paths. This method involves utilizing the generated paths to approximate the no-arbitrage European call option prices. By treating the generated paths as if they were simulated via the Monte Carlo method, the no-arbitrage option price can be estimated by averaging the discounted payoffs from each path. As discussed in Section 2.3, this option pricing method requires simulating the synthetic training set with a drift rate equal to the risk-free rate. Consequently, if the generated paths are accurately learned, they should exhibit drift rates that are close if not equal to the risk-free rate, thereby aligning with the assumptions of the risk-neutral valuation framework. The option prices derived through this process are referred to as the 'generated prices'.

In Section 2.3, we also explored the approaches for obtaining the theoretical no-arbitrage option price when asset price follows GBM or the Heston model. Comparing the generated and theoretical prices offers insights into the accuracy of the generated paths in reflecting the original asset price dynamics.

However, this approach is not applicable when training sets consist of actual market data, as the learned drifts from actual market prices do not correspond to the market risk-free rate. As a result, the expected payoff derived from such paths would not represent no-arbitrage option prices. Therefore, any comparison between these generated option prices and actual market option prices would not be meaningful.

Chapter 5

Results

This chapter evaluates the efficacy of the DDPM in generating synthetic asset price paths to mirror different training set dynamics. We organize the discussion into three sections, each corresponding to a different construction methodology of the training sets: 1) GBM simulated training sets, 2) Heston model simulated training sets, and 3) market data bootstrap training sets. In each section, we will assess the DDPM's performance based on 10,000 generated asset price paths using the evaluation criteria outlined in Chapter 4.

5.1 GBM Simulated Training Set

In this section, we examine DDPM's performance in generating asset price paths when the original paths' dynamics are characterized by GBM. To do this, we consider training sets simulated to follow GBM at different volatility levels, while keeping the other parameters the same (as shown in Table 5.1). Specifically, we examine three volatility scenarios: low volatility ($\sigma = 10\%$), medium volatility ($\sigma = 30\%$), and high volatility ($\sigma = 50\%$).

Parameter	S_0	T	μ
Value	100	0.5	0.05

Table 5.1: GBM - default parameters that are held constant for simulating all three volatility scenarios.

Qualitative Analysis

In Figure 5.1, for each volatility scenario, we illustrate five randomly selected paths from the training set and five randomly generated paths from the trained DDPM. A visual comparison indicates that across all three scenarios, the generated paths closely mirror the dynamics of the original paths by reflecting similar ranges of volatility, with no apparent issues at the left or right boundaries.

KS-Statistic

By employing the KS-statistic, we analyze if the generated paths' log returns distribution is similar to that of the original paths. Table 5.2 summarizes the KS-statistic values for each volatility scenario. Notably, all the values are below 0.05, indicating a close alignment in the distribution of log returns between the generated and original paths across all three volatility scenarios.

	$\sigma = 0.1$	$\sigma = 0.3$	$\sigma = 0.5$
KS-statistic value	0.0247	0.0256	0.0291

Table 5.2: GBM - KS-statistic value summary.

Drift and Volatility Analysis

Next, we qualitatively compare the distributions of drift and volatility between the generated and original paths' log returns for each volatility scenario based on Figure 5.2. The drift distributions exhibit a high degree of congruence. The volatility distributions reveal underestimation in the generated paths compared to the original. Such deviation is the most apparent in the high volatility scenario, which also shows heavier tails in generated paths when compared to the original. Despite these discrepancies, the general resemblance between the shapes of the distributions indicates that the DDPM effectively captures the inherent randomness present in the original paths. Rather than generating a uniform set of paths characterized by identical drift and volatility, the DDPM demonstrates its capability to produce a diverse array of paths. Each path exhibits unique variations, yet collectively, they closely represent the underlying dynamics.

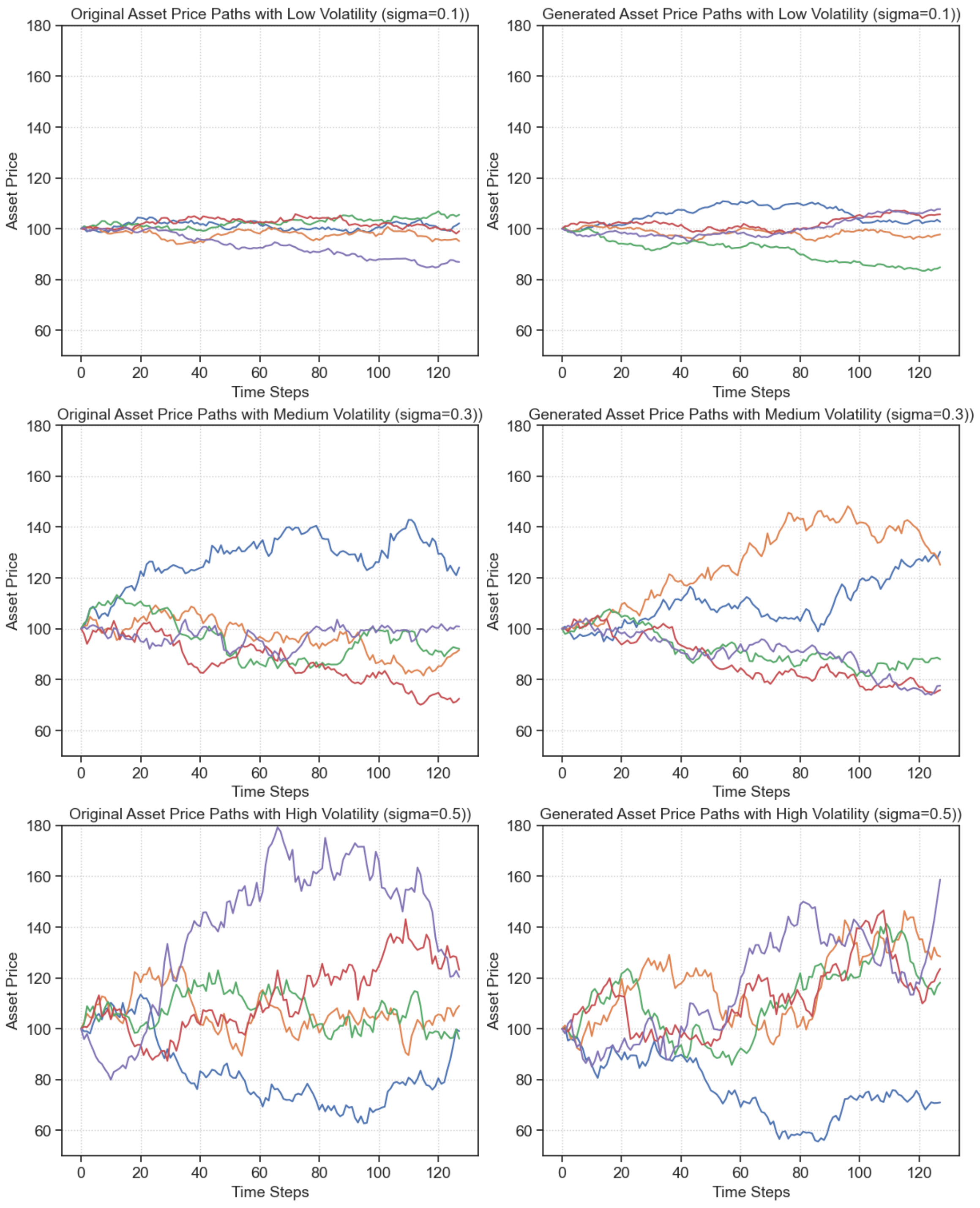

Figure 5.1: GBM - original paths vs. generated paths.

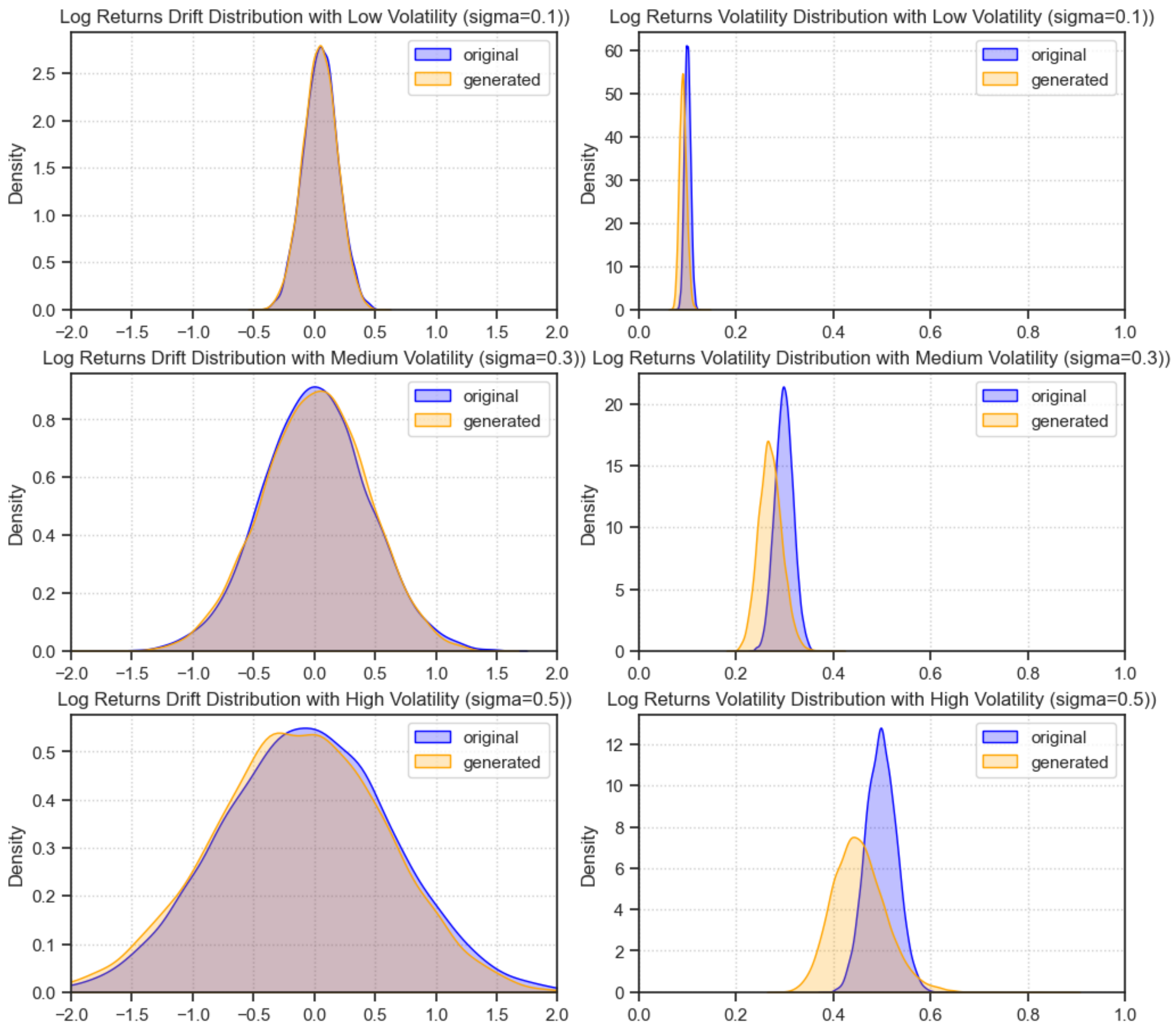

Figure 5.2: GBM - log returns drift and volatility distribution.

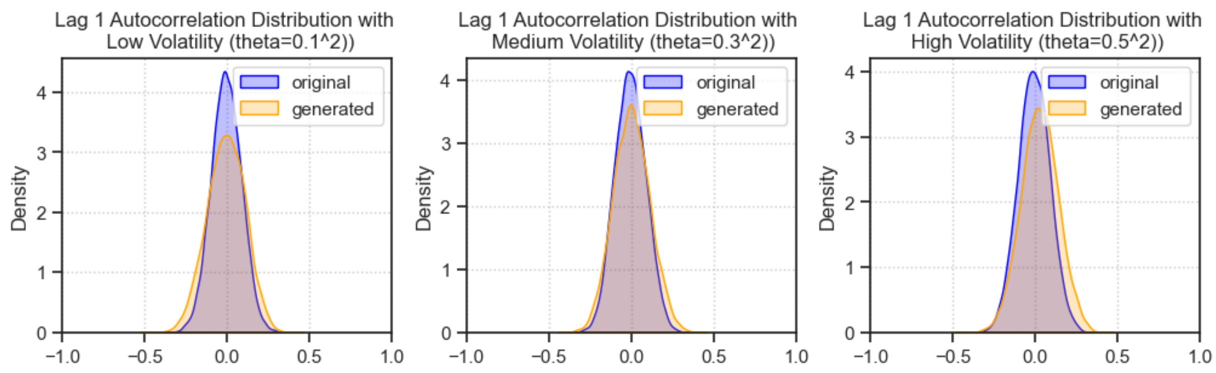

Figure 5.3: GBM - log returns lag 1 autocorrelation values distribution.

Autocorrelation Analysis

We then proceed to analyze the autocorrelation of log returns for both the generated and original paths within each volatility scenario. This examination involves assessing the distributions of autocorrelation values from lag 1 to 20. As depicted in Figure 5.3, the distribution of lag 1 autocorrelation values is provided as a representative example. Notably, the autocorrelation values across all examined lags predominantly cluster around 0, indicating a lack of autocorrelation for the majority of paths in the range of lag 1 to 20. This pattern is consistently observed in both the generated and original paths for all three volatility scenarios, suggesting a close alignment in their autocorrelation structures.

Option Pricing Analysis

Lastly, we perform an option pricing analysis to determine if the generated paths effectively mirror the original paths' dynamics, specifically in the context of approximating no-arbitrage prices for European call options. We assume that the risk-free rate equals to the drift rate used to simulate the training set, which is 5%. For a given strike price K, we employ the approach outlined in Section 4.2 to estimate the no-arbitrage prices for European call options, denoted as the 'generated prices'. Correspondingly, the 'theoretical prices' are derived using the Black-Scholes formula (2.6), (2.7), and (2.8).

Table 5.3 displays the generated prices for a range of strike prices K, along with the theoretical prices. Using the notation from Section 2.3, we categorize different call options based on K into ITM, ATM and OTM options. We start by examining the absolute average relative error for each option type. Across all three volatility levels, the generated prices for

$\sigma = 0.1$							
Option type	ITM			ATM	OTM		
Strike price	70	80	90	100	110	120	130
Theoretical price	31.7283	21.9756	12.3068	4.1923	0.6166	0.0348	0.0008
Generated price	31.6375	21.8845	12.2138	4.0966	0.5613	0.0218	0.0003
Relative error	-0.29%	-0.41%	-0.76%	-2.28%	-8.96%	-37.36%	-57.46%
Absolute avg. relative error	0.49%			2.28%	34.59%		
$\sigma = 0.3$							
Option type	ITM			ATM	OTM		
Strike price	70	80	90	100	110	120	130
Theoretical price	31.9779	23.0891	15.4860	9.6349	5.5871	3.0441	1.5728
Generated price	31.9808	23.1429	15.5662	9.6964	5.5836	2.9884	1.4994
Relative error	0.01%	0.23%	0.52%	0.64%	-0.06%	-1.83%	-4.67%
Absolute avg. relative error	0.25%			0.64%	2.19%		
$\sigma = 0.5$							
Option type	ITM			ATM	OTM		
Strike price	70	80	90	100	110	120	130
Theoretical price	33.7942	26.3842	20.1600	15.1272	11.1841	8.1723	5.9173
Generated price	31.0170	23.8428	17.9220	13.1771	9.4986	6.7180	4.6874
Relative error	-8.22%	-9.63%	-11.10%	-12.89%	-15.07%	-17.79%	-20.78%
Absolute avg. relative error	9.65%			12.89%	17.88%		

Table 5.3: GBM - theoretical no-arbitrage prices and generated prices for different strike prices.

ITM and ATM options have higher accuracy than those for OTM options. For instance, at $\sigma = 0.1$, the absolute average relative errors for ITM and ATM options are 0.49% and 2.28% respectively, in contrast to the 34.59% error for OTM options. This difference can be attributed to the smaller number of positive payoffs at higher strike prices. This issue becomes particularly pronounced in the low volatility scenario, which has the least number of paths yielding positive payoffs. While increasing the number of generated paths could improve accuracy, computational constraints limit this study to $10,000$ paths. In typical Monte Carlo methods for option pricing, it is common to use a significantly larger set of paths (e.g., 1 million paths) to ensure accuracy.

By comparing the different volatility scenarios, it is evident that the medium volatility scenario exhibits the lowest absolute average relative error: 0.25% for ITM options, 0.64% for ATM options, and 2.19% for OTM options. The low volatility case shows good accuracy overall, except for the OTM options. In addition to the previously discussed reason of having a small number of paths yielding positive payoffs, this can also be attributed to the relatively small magnitudes of both theoretical and generated prices for these OTM options (less than one dollar). With small magnitudes of prices, even minor discrepancies can result in significant relative errors. In the high volatility scenario, we see an overall high level of relative errors. This is attributable to the greater divergence in volatility distribution between the generated and original paths in the high volatility scenario, as depicted in Figure 5.2.

Overall, our option pricing analysis suggests that the DDPM is more effective at capturing the GBM dynamics in the lower volatility scenarios. Although the intent is not to utilize the generated paths for option pricing, rather to assess the quality of the generated paths, the low relative errors observed in low and medium volatility scenarios, especially for ITM and ATM options, signal a close alignment between the dynamics of the generated and original paths.

In summary, our evaluation demonstrates that the DDPM is a proficient tool for synthesizing asset price paths within the GBM framework, showing particular effectiveness in the low and medium volatility scenarios.

5.2 Heston Model Simulated Training Set

In this section, we analyze DDPM's capability in synthesizing asset price paths that capture the Heston model dynamics, which introduces additional complexity compared to the GBM by incorporating stochastic volatility. In alignment with our GBM setup, we construct training sets representing three different volatility scenarios by altering the long-term variance θ while holding other parameters constant (as detailed in Section 2.2.2). The default parameter settings for the Heston model can be found in Table 5.4, and the θ values considered are 0.1^2, 0.3^2, and 0.5^2, corresponding to low, medium, and high volatility scenarios respectively. It is noteworthy that θ is a variance measure, hence $\sqrt{\theta}$ provides a volatility measure analogous σ in GBM.

Parameter	S_0	T	μ	v_0	κ	ξ	ρ
Value	100	0.5	0.05	0.1^2	1	0.1	0.5

Table 5.4: Heston model - default parameters that are held constant for simulating all three volatility scenarios.

Qualitative Analysis

We begin with a visual comparison of five original and five generated paths for each volatility scenario (see Figure 5.4). Similar to the GBM scenarios, the boundary behaviors do not present any noticeable issues. In addition, given the initial variance setting of $v_0 = 0.1^2$, the original paths in the medium and high volatility scenarios ($\theta = 0.3^2$ and $\theta = 0.5^2$) demonstrate increased volatility level through time. The generated paths mirror this dynamics effectively, with the paths at later timesteps displaying a higher degree of variation.

KS-Statistic

In Table 5.5, we present the KS-statistic values for assessing the similarity between the log returns distributions of the generated and original paths. All values fall below 0.05, confirming minimal deviation between these distributions.

	$\theta = 0.1^2$	$\theta = 0.3^2$	$\theta = 0.5^2$
KS-statistic value	0.0238	0.0221	0.0301

Table 5.5: Heston model - KS-statistic value summary.

Drift and Volatility Analysis

Figure 5.5 offers a visual comparison between original and generated paths' log returns drift and volatility distributions. The drift distributions between the original and generated paths show no significant deviations. In line with our findings from the GBM scenarios, the volatility distributions indicate a slight underestimation in the generated paths' volatility, particularly noticeable when $\theta = 0.5^2$. Moreover, when $\theta = 0.1^2$, the generated paths exhibit a volatility distribution with lighter tails compared to those in the original paths. Despite these discrepancies, the DDPM overall successfully generates paths with a range of individual drifts and volatilities, collectively forming distributions that closely mirror those of the original paths.

Autocorrelation Analysis

Our autocorrelation analysis suggests that the distributions of autocorrelation values for the generated paths closely mirror those of the original paths across lags 1 to 20. Figure

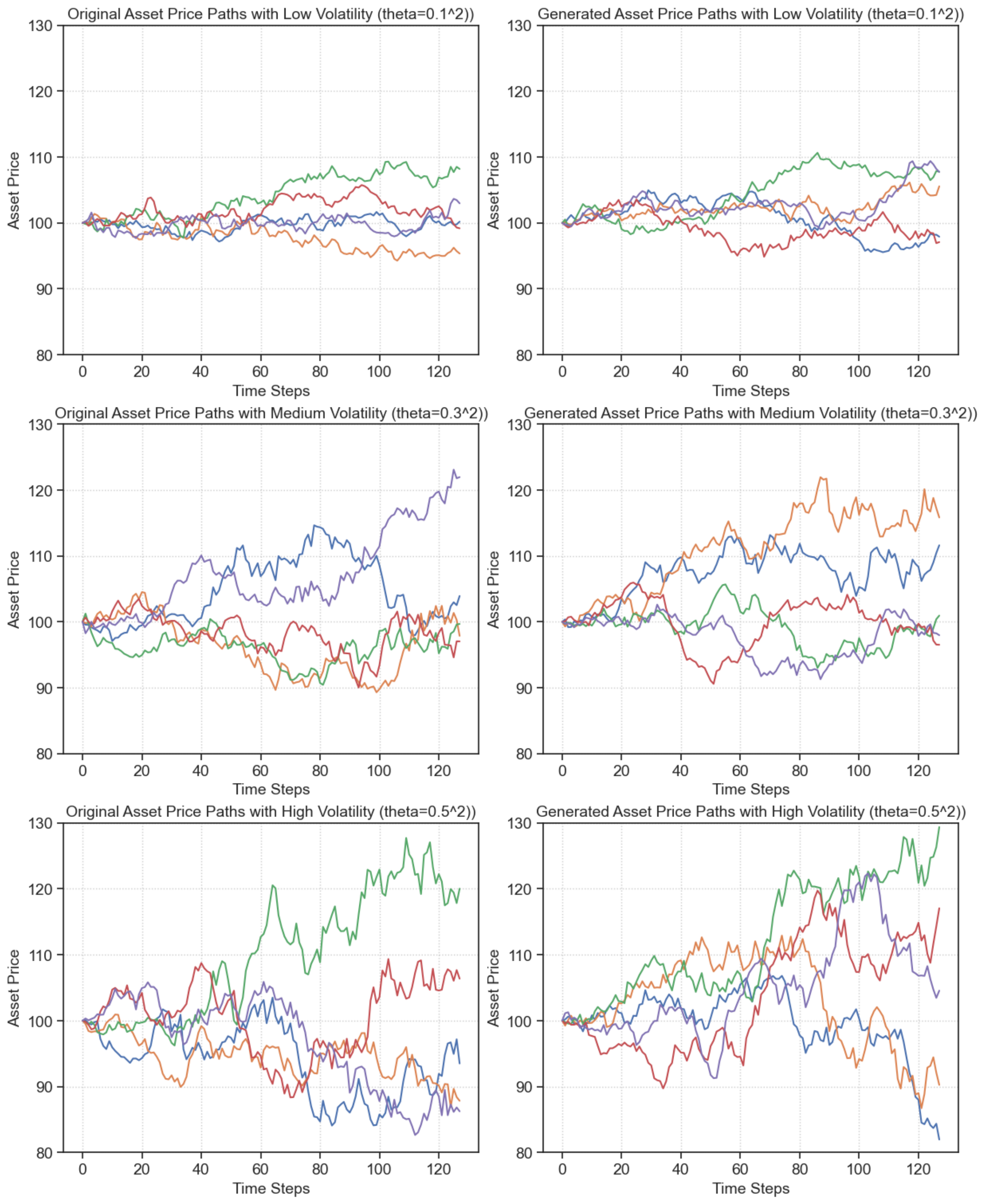

Figure 5.4: Heston model - original paths vs. generated paths.

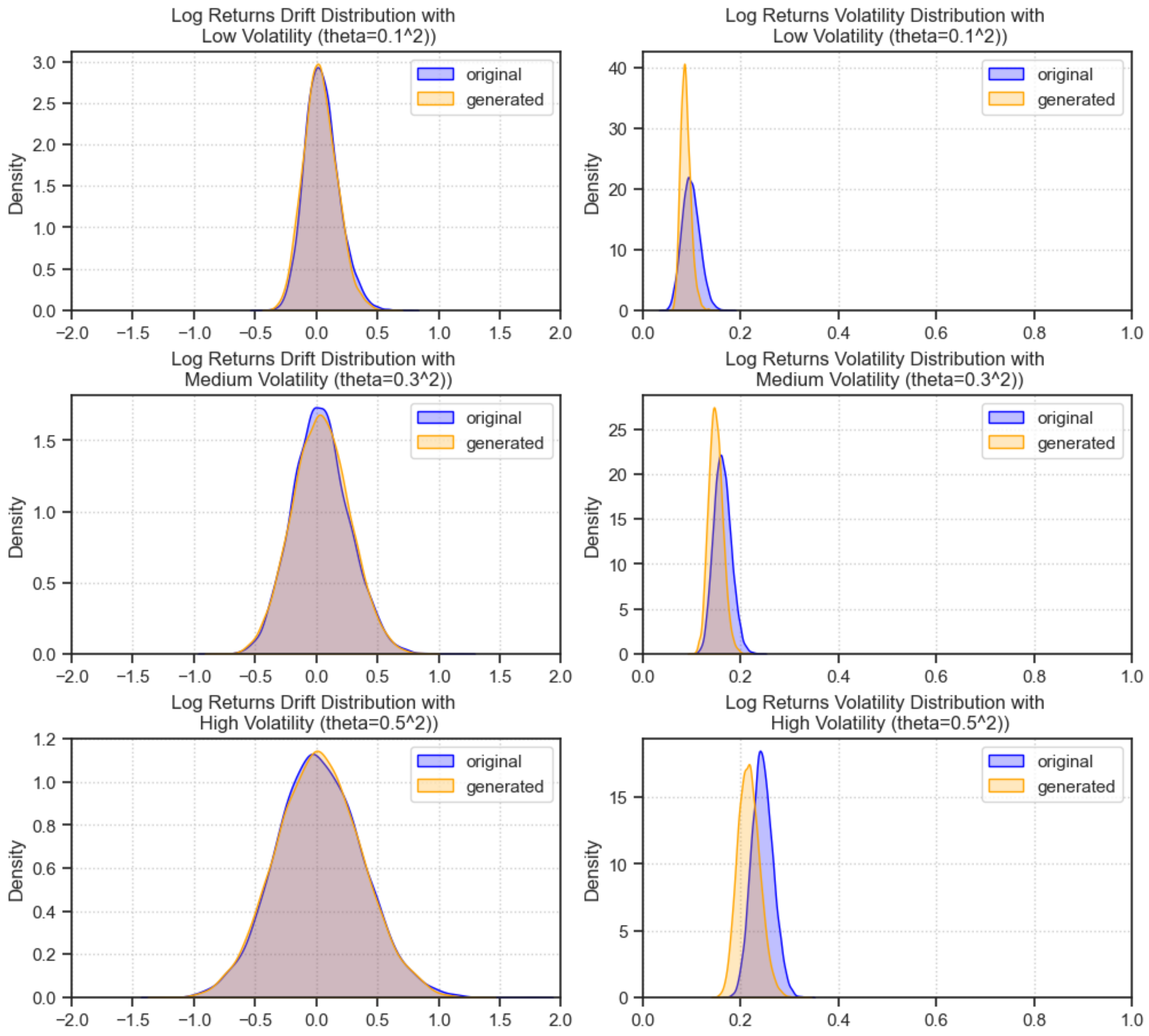

Figure 5.5: Heston model - log returns drift and volatility distribution.

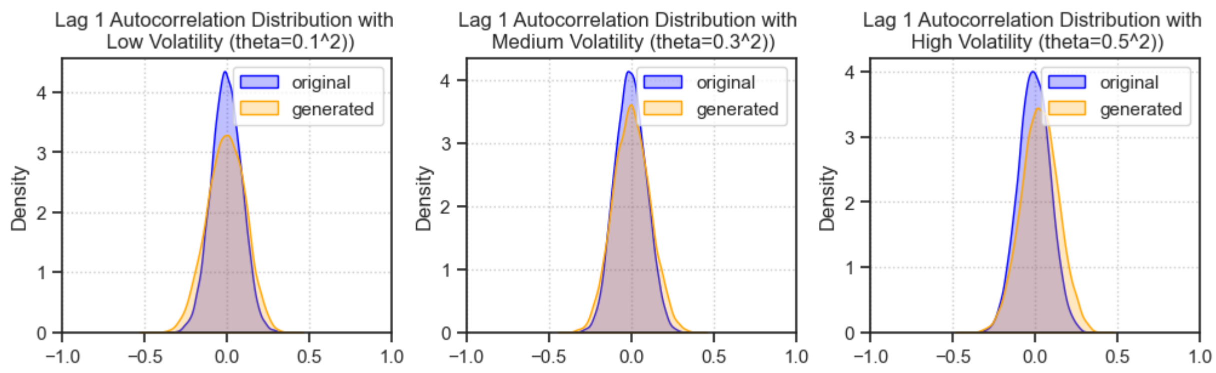

Figure 5.6: Heston model - log returns lag 1 autocorrelation values distribution.

5.6 provides an illustration for the distribution of lag 1 autocorrelation values. This resemblance indicates that the generated paths maintain autocorrelation structures similar to those of the original paths.

Option Pricing Analysis

Option pricing analysis offers additional insights into how effectively the generated paths replicate Heston model dynamics by assessing the accuracy of the resulting option prices. We assume a risk-free rate of 5%, aligning it with the drift rate used in the Heston model training set simulations. The generated prices are computed using the method described in Section 4.2, and we compare them with their theoretical counterparts[1].

Table 5.6 displays both the theoretical and generated prices for a range of strike prices across varying volatility levels. Consistent with our findings from the GBM scenario discussed in Section 5.1, the highest accuracy in generated prices is observed for ITM options, followed by ATM and then OTM options. This pattern is largely due to the reduced number of paths with positive payoffs at higher strike prices, consequently affecting accuracy. Additionally, significant errors tend to occur when both theoretical and generated prices are relatively low. For example, with $\theta = 0.1^2$ and $K = 130$, the theoretical price is 0.0113 and generated price is 0.0028, leading to a relative error of -74.82%. This is not surprising when dealing with small price magnitudes, where even minor absolute discrepancies

[1]The theoretical option prices under the Heston model are approximated using a Monte Carlo simulation with 10^7 paths, providing two-digit accuracy (see Section 2.3 for details). The parameters for this simulation align with those used for the Heston model training set, ensuring consistency in the comparison.

can result in large relative errors. Similar to the GBM scenario, increasing the number of generated paths could improve the accuracy in generated prices.

$\theta = 0.1^2$							
Option type	ITM			ATM	OTM		
Strike price	70	80	90	100	110	120	130
Theoretical price	31.7315	21.9785	12.2667	4.0939	0.7416	0.0966	0.0113
Generated price	30.9191	21.1660	11.4658	3.5412	0.5030	0.04089	0.0028
Relative error	-2.56%	-3.70%	-6.53%	-13.50%	-32.18%	-57.68%	-74.82%
Absolute avg. relative error	4.26%			13.50%	54.89%		
$\theta = 0.3^2$							
Option type	ITM			ATM	OTM		
Strike price	70	80	90	100	110	120	130
Theoretical price	31.7238	22.0057	12.8390	5.8684	2.0961	0.6169	0.1596
Generated price	31.6945	21.9788	12.8639	5.8884	2.0320	0.5312	0.1053
Relative error	-0.09%	-0.12%	0.19%	0.34%	-3.06%	-13.89%	-34.04%
Absolute avg. relative error	0.13%			0.34%	17.00%		
$\theta = 0.5^2$							
Option type	ITM			ATM	OTM		
Strike price	70	80	90	100	110	120	130
Theoretical price	31.7785	22.4451	14.2948	8.1578	4.2109	2.0015	0.8927
Generated price	31.6164	22.2991	14.1640	8.0051	4.0463	1.8435	0.7657
Relative error	-0.51%	-0.65%	-0.92%	-1.87%	-3.91%	-7.89%	-14.23%
Absolute avg. relative error	0.69%			1.87%	8.68%		

Table 5.6: Heston model - theoretical no-arbitrage prices and generated prices for different strike prices.

A comparison across different volatility levels reveals that the medium volatility case has the highest accuracy for both ITM and ATM options, followed by the high volatility case, and then the low volatility case. This can be explained by our previous findings from Figure 5.5, where the medium volatility case shows a more precise alignment in the volatility distribution of the generated and original paths, compared to the less accurate alignment in the high and low volatility cases. Interestingly, the high volatility scenario in the Heston model show significantly lower absolute relative errors (0.69%, 1.87% and 8.68% for ITM, ATM and OTM options) than what is observed in the GBM's high volatility scenario (9.65%, 12.89% and 17.88% for ITM, ATM and OTM options). Although this seems like a contradiction, it can be explained by the Heston model's time-varying volatility, which starts at an initial level of $\sqrt{v_0} = 0.1$ and gradually converges to $\sqrt{\theta} = 0.5$. Consequently, the majority of paths average out to a volatility range between 0.2 to 0.3, as depicted in Figure 5.5. This characteristic makes the high volatility scenario under the Heston model more analogous to the medium volatility scenario under the GBM, where a similarly high accuracy level is observed.

Overall, the low absolute relative errors for ITM and ATM options demonstrate DDPM's capability in generating asset price paths that capture the Heston model's dynamics. The comparatively higher errors noted for OTM options suggest potential for improvement through the generation of additional paths.

To summarize, our analysis suggests that the DDPM-generated paths effectively mimic the Heston model's dynamics across various volatility levels. This indicates that the DDPM can be a valuable method for producing synthetic asset prices that reflect more realistic price dynamics like time-varying volatility.

5.3 Market Data Bootstrap Training Set

In this section, we explore DDPM's effectiveness in capturing real market data dynamics by analyzing three stocks: Procter & Gamble (PG), Apple (AAPL), and Tesla (TSLA). We consider the historical price data for each of these stocks from January 4, 2021, to October 25, 2023, and we provide an illustration of these data in Figure 5.7. We perform our study under the assumption that the price dynamics of these stocks remain stable during this period. The historical annualized volatility of these stocks during this period, which are 17.63%, 28.34%, and 59.79%, make them suitable proxies for low, medium, and high volatility scenarios. Applying Algorithm 3, we construct a training set of bootstrap asset price paths for each stock[2].

Note that with real market data, we cannot evaluate the quality of generated paths using 'generated' option prices as we do with the GBM and Heston model data for reasons explained in Section 4.2. Therefore, we rely on the rest of the metrics discussed in Chapter 4 to assess the quality of the DDPM-generated paths.

Qualitative Analysis

We begin by comparing five original bootstrap paths against five generated paths for each stock, as shown in Figure 5.8. To facilitate comparison, we standardize the asset price range for all stocks to span from $S_0 - 100$ to $S_0 + 100$. This standardization allows us to easily

[2]The initial stock prices S_0 are set to the closing prices on January 4, 2021, which are \$137.82 for PG, \$129.41 for AAPL, and \$243.26 for TSLA.

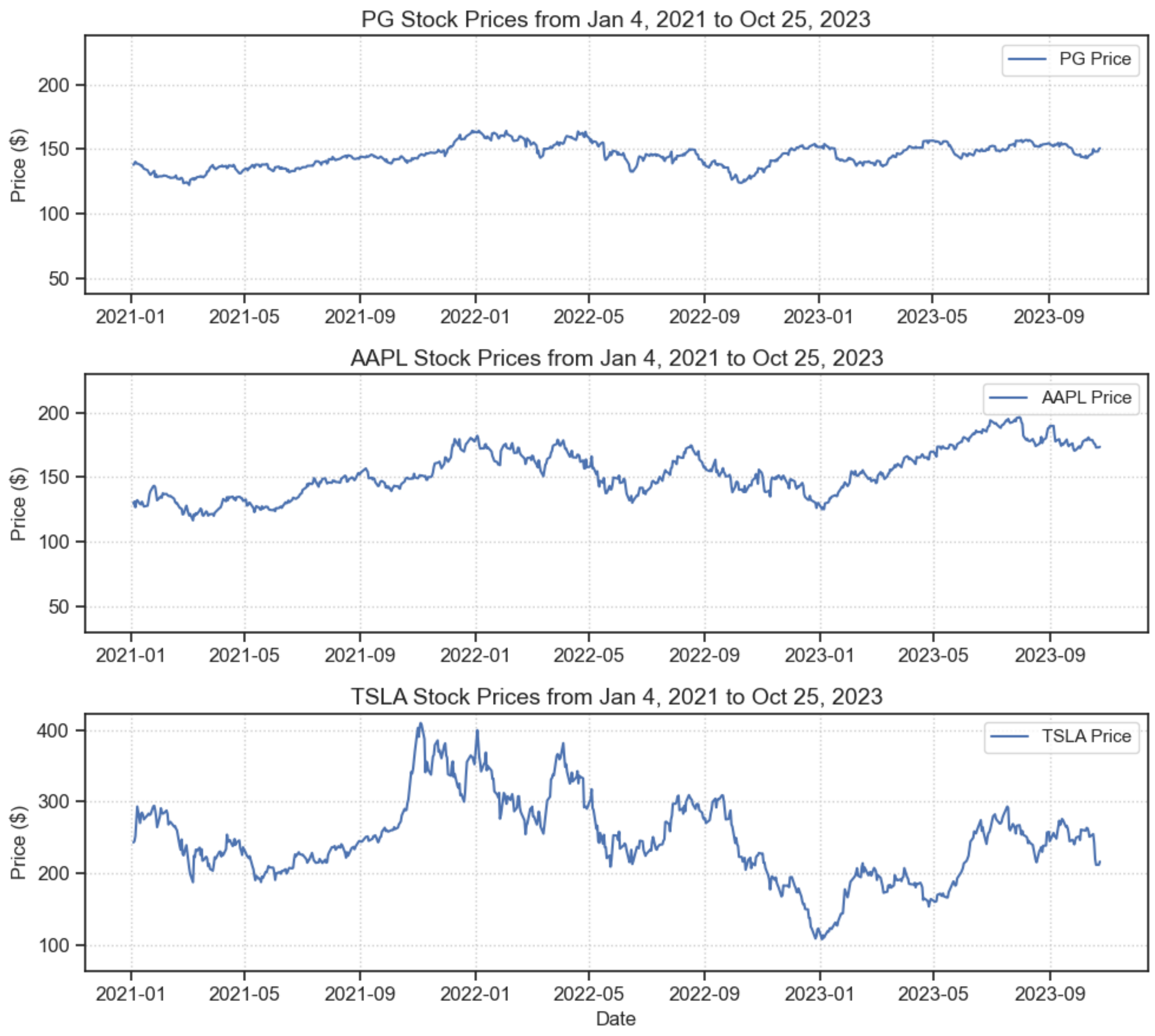

Figure 5.7: Market historical price data.

compare their volatility levels despite different initial prices. Visually, the generated paths closely mirror the original ones, with PG and AAPL exhibit lower variations in comparison to TSLA. Additionally, no notable issues are observed in the boundary behavior of these paths.

KS-Statistic

The KS-statistic values presented in Table 5.7 are all below 0.05, indicating a close alignment in the distributions of log returns between the generated and original paths for all three stocks.

	PG	AAPL	TSLA
KS-statistic value	0.0391	0.0310	0.0346

Table 5.7: Market bootstrap data - KS-statistic value summary.

Drift and Volatility Analysis

We conduct an analysis of the log returns drift and volatility distributions based on Figure 5.9. The drift distributions of the generated paths exhibit a high degree of resemblance with those of the original paths. This is particularly evident in the case of TSLA, where the slight rightward skew in the original paths' drift distribution is accurately mirrored in the generated paths. The examination of the volatility distributions indicate an overall close alignment between the generated and original paths, with a slight underestimation observed in the case of TSLA.

Autocorrelation Analysis

We conduct an analysis on log returns autocorrelation values for lags 1 to 20 for all three stocks. Figure 5.10 illustrates the distribution of these values at lag 1 as a representative example. Echoing our observations from the synthetic training sets, the log returns autocorrelation values for all examined lags display similar distributions between the original and generated paths, predominantly clustering around zero. This similarity indicates a consistent autocorrelation structure in the log returns of both the original and generated paths.

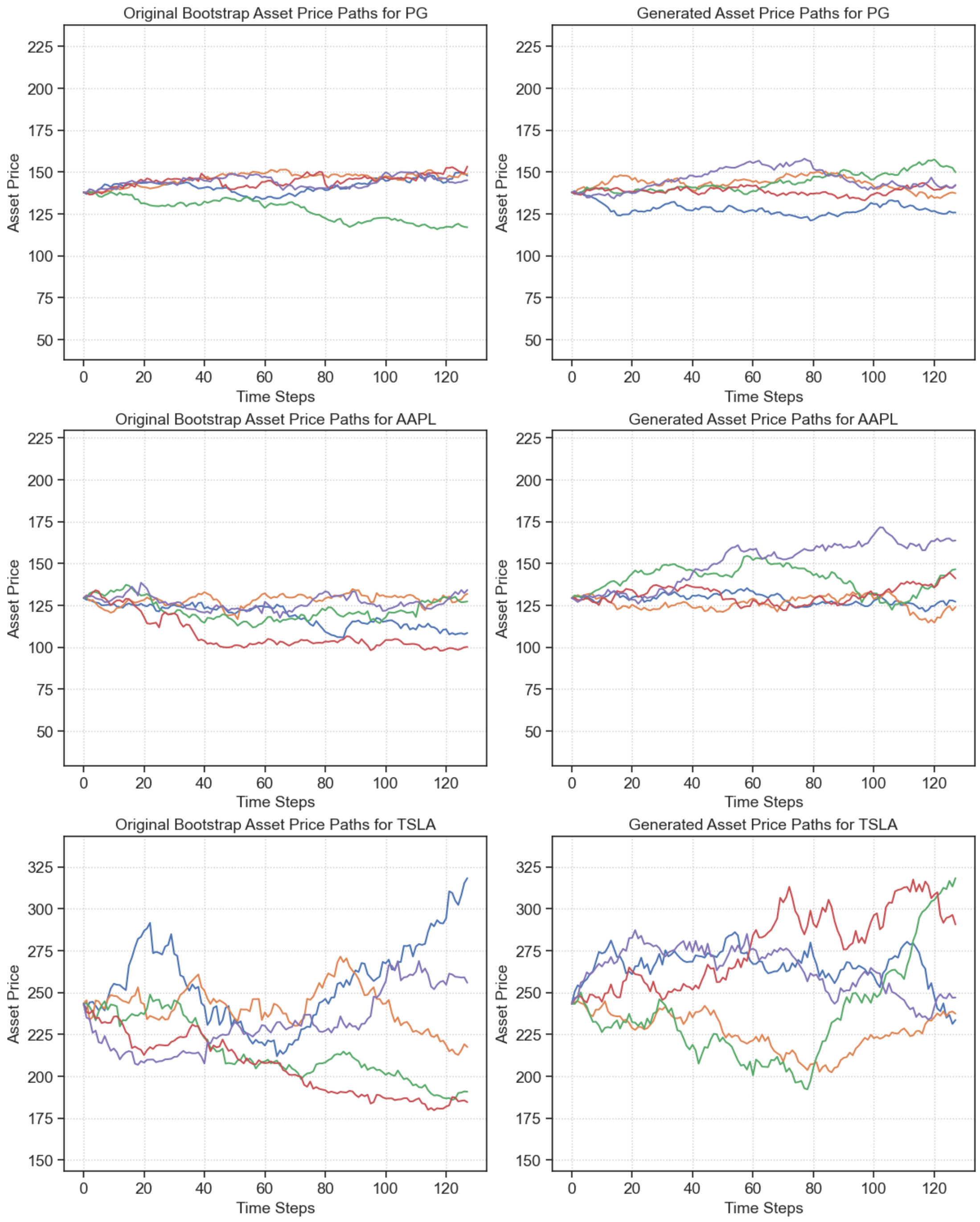

Figure 5.8: Market bootstrap data - original paths vs. generated paths.

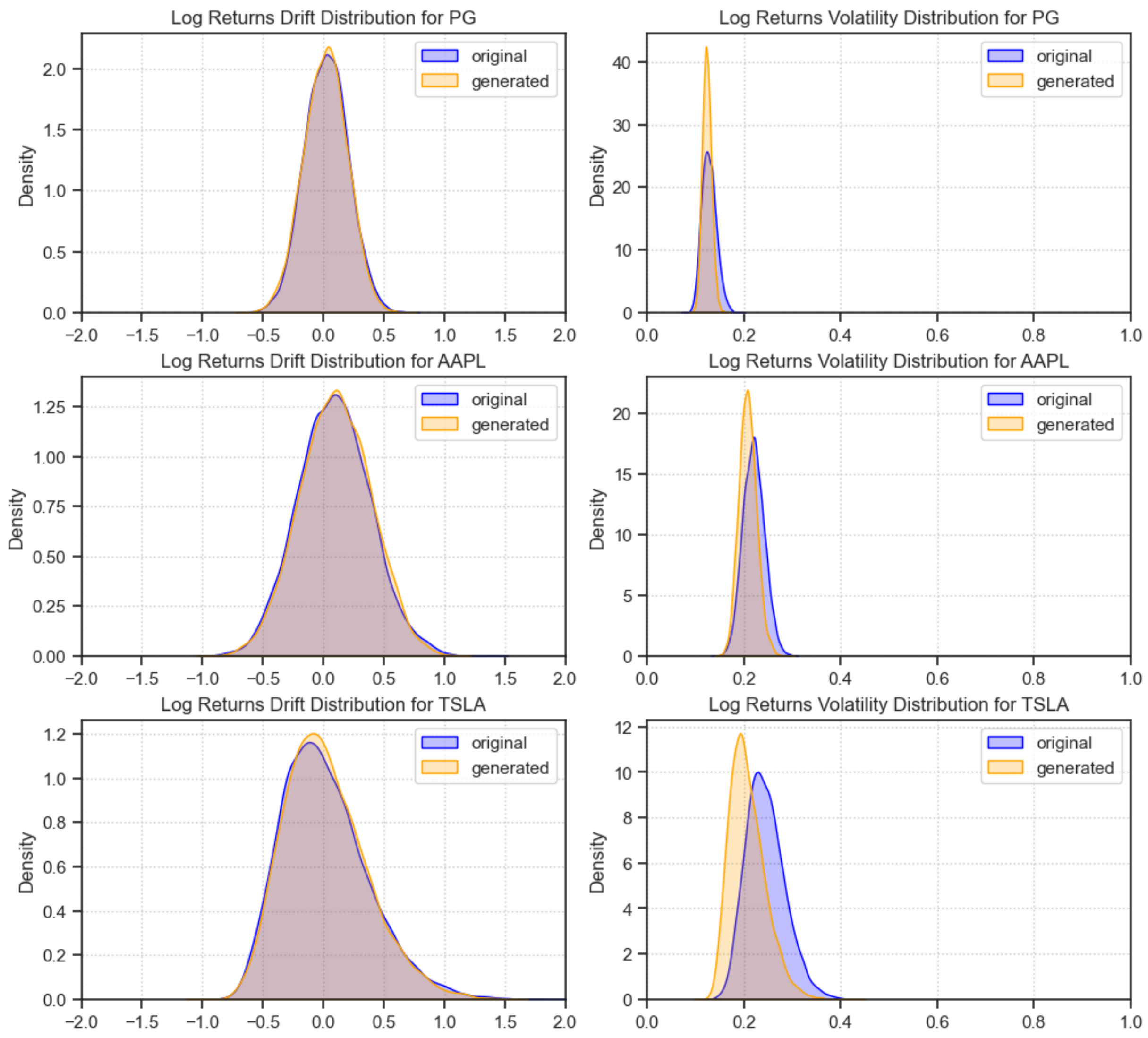

Figure 5.9: Market bootstrap data - log returns drift and volatility distribution.

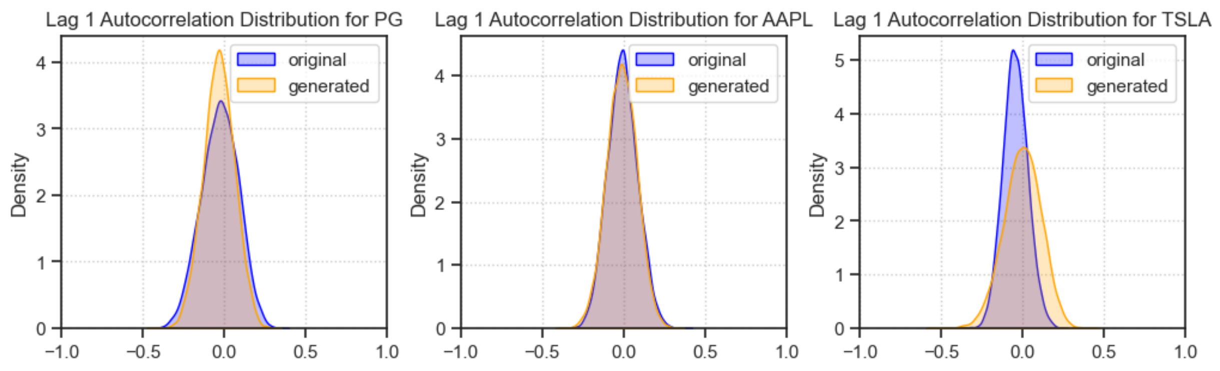

Figure 5.10: Market bootstrap data - log returns lag 1 autocorrelation values distribution.

In conclusion, our analysis suggests that the DDPM is capable of generating synthetic asset price paths that mirror the diverse volatility dynamics of the selected stocks, demonstrating the model's potential for practical applications in financial markets.

Chapter 6

Conclusion

In this study, we have introduced a novel method for generating synthetic financial asset price paths using the DDPM. Our approach uniquely incorporates a DCT in preprocessing the training data. This technique shifts the learning process from the time domain into the frequency domain, significantly enhancing the DDPM's performance. Our method can generate synthetic paths that resemble the dynamics of the original paths, eliminating the need for explicit assumptions about the model form of the original paths' dynamics and avoiding the traditional calibration process.

We evaluated the DDPM's efficacy under various volatility scenarios using synthetic training data based on GBM and Heston model and bootstrap market training data. The analysis revealed the DDPM-generated paths qualitatively closely resemble the original paths. From a quantitative perspective, the statistical characteristics of the log returns from the generated paths showed a strong alignment with those of the original paths. This is evidenced by the KS-statistic values which fall under 0.05 for all examined scenarios, as well as the closely aligned distributions for drift and autocorrelation values bewteen the original and generated paths. In addition, with synthetic training data, when applying the generated paths to approximate the no-arbitrage European call option prices, we obtained high level of accuracy for certain volatility and option types. In particular, with $\sigma = 30\%$ for GBM and $\theta = 30\%$ for Heston model, the approximated option prices incurred less than 1% absolute relative errors for ITM and ATM options. These findings confirm the DDPM's effectiveness in synthetic asset price generation.

Despite its effectiveness, our DDPM model comes with limitations. One limitation is the underestimation of volatility levels in our generated paths, particularly in cases where original paths exhibit high volatility (e.g., $\sigma = 50\%$ for GBM training set). Future research can explore potential methods for refining the model to more closely capture the high volatility dynamics.

Another challenge lies in the requirement for a substantial training dataset with consistent underlying dynamics. This is less of an issue with synthetic data, as we can generate a large number of synthetic samples under the same price dynamic. However, with real market data, we relied on a bootstrapping technique for sampling additional training data. This technique may distort the structure of the original data, potentially creating a training set that does not accurately represent the dynamics of the original series. For instance, if the market data has volatility dynamics similar to the Heston model, which gradually transition from one level to another, our bootstrapping approach cannot retain such dynamics in the resampled data. This presents another area for future research: developing methods to refine the construction of the training dataset. Such development would ensure that the training paths consistently represent the specific dynamics we aim for the DDPM to learn, thereby enhancing the model's applicability and accuracy in replicating complex market dynamics.

References

[1] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[2] Samuel A. Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E. Tillman, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: Opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, ICAIF '20, New York, NY, USA, 2021. Association for Computing Machinery.

[3] J. F. Barrett and D. J. Wright. The random nature of stock-market prices. *Operations Research*, 22(1):175–177, 1974.

[4] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

[5] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7327–7347, November 2022.

[6] Ka Chan, C. Lenard, and Terence Mills. An introduction to Markov chains. In *Proceedings of the MAV 49th Annual Conference*, La Trobe University, Bundoora, VIC, Australia, 12 2012.

[7] Carl Chiarella, Xue-Zhong He, and Christina Sklibosios Nikitopoulos. Stochastic processes for asset price modelling. In *Derivative Security Pricing: Techniques, Methods and Applications*, pages 7–36. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[8] Jean Baptiste Joseph Fourier. *The analytical theory of heat*. Cambridge Library Collection - Mathematics. Cambridge University Press, reprint edition, 2009.

[9] Federico Gatta, Fabio Giampaolo, Edoardo Prezioso, Gang Mei, Salvatore Cuomo, and Francesco Piccialli. Neural networks generative models for time series. *Journal of King Saud University - Computer and Information Sciences*, 34(10, Part A):7920–7939, 2022.

[10] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3313–3332, 2023.

[11] Wolfgang Härdle, Joel Horowitz, and Jens-Peter Kreiss. Bootstrap methods for time series. *International Statistical Review / Revue Internationale de Statistique*, 71(2):435–459, 2003.

[12] Martin K. Hess. What drives Markov regime-switching behavior of stock markets? The Swiss case. *International Review of Financial Analysis*, 12(5):527–543, 2003.

[13] Steven L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The Review of Financial Studies*, 6(2):327–343, 1993.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, Vancouver, Canada, 2020.

[15] James Hurst, Kirill Mayorov, and Joseph Francois Tagne Tatsinkou. The generation of synthetic data for risk modelling. *Journal of Risk Management in Financial Institutions*, 15(3):260–269, 2022.

[16] Kiyosi Itô. Stochastic integral. *Proceedings of the Imperial Academy*, 20(8):519 – 524, 1944.

[17] Abdul Jabbar, Xi Li, and Bourahla Omar. A survey on generative adversarial networks: Variants, applications, and training. *ACM Comput. Surv.*, 54(8), oct 2021.

[18] George Jabbour and Yi-Kang Liu. Option pricing and Monte Carlo simulations. *Journal of Business Economics Research (JBER)*, 3, 02 2011.

[19] Zuzana Janková. Drawbacks and limitations of Black-Scholes model for options pricing. *Journal of Financial Studies and Research*, 2018:1–7, 08 2018.

[20] Ying Jiao, Chunhua Ma, Simone Scotti, and Chao Zhou. The alpha-Heston stochastic volatility model. *Mathematical Finance*, 31:943–978, 2021.

[21] Herb Johnson and David Shanno. Option pricing when the variance is changing. *The Journal of Financial and Quantitative Analysis*, 22(2):143–151, 1987.

[22] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[23] Pengzhi Li, Yan Pei, and Jianqiang Li. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, 138:110176, 2023.

[24] A. G. Malliaris. Wiener process. In *Econometrics*, pages 276–278. Palgrave Macmillan UK, London, 1990.

[25] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021. arXiv preprint arXiv:2102.09672.

[26] Jonas Oppenlaender. The creativity of text-to-image generation. In *Proceedings of the 25th International Academic Mindtrek Conference*, Academic Mindtrek '22, page 192–202, New York, NY, USA, 2022. Association for Computing Machinery.

[27] B.K. Pagnoncelli, D. Ramírez, H. Rahimian, et al. A synthetic data-plus-features driven approach for portfolio optimization. *Computational Economics*, 62:187–204, 2023.

[28] Emanuel Parzen. *Stochastic Processes.* Dover Publications, Mineola, New York, United States, 2015.

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[31] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[32] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and

David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR.

[33] Gilbert Strang. Wavelets. *American Scientist*, 82(3):250–255, 1994.

[34] Alexander van Haastrecht and Antoon Pelsser. Efficient, almost exact simulation of the Heston stochastic volatility model. *International Journal of Theoretical and Applied Finance*, 13(01):1–43, 2010.

[35] James Xiong and Thomas Idzorek. The impact of skewness and fat tails on the asset allocation decision. *Financial Analysts Journal*, 67, 04 2011.

[36] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4), nov 2023.

[37] Jinsung Yoon, William Zame, and Mihaela Schaar. Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Transactions on Biomedical Engineering*, PP, 11 2017.