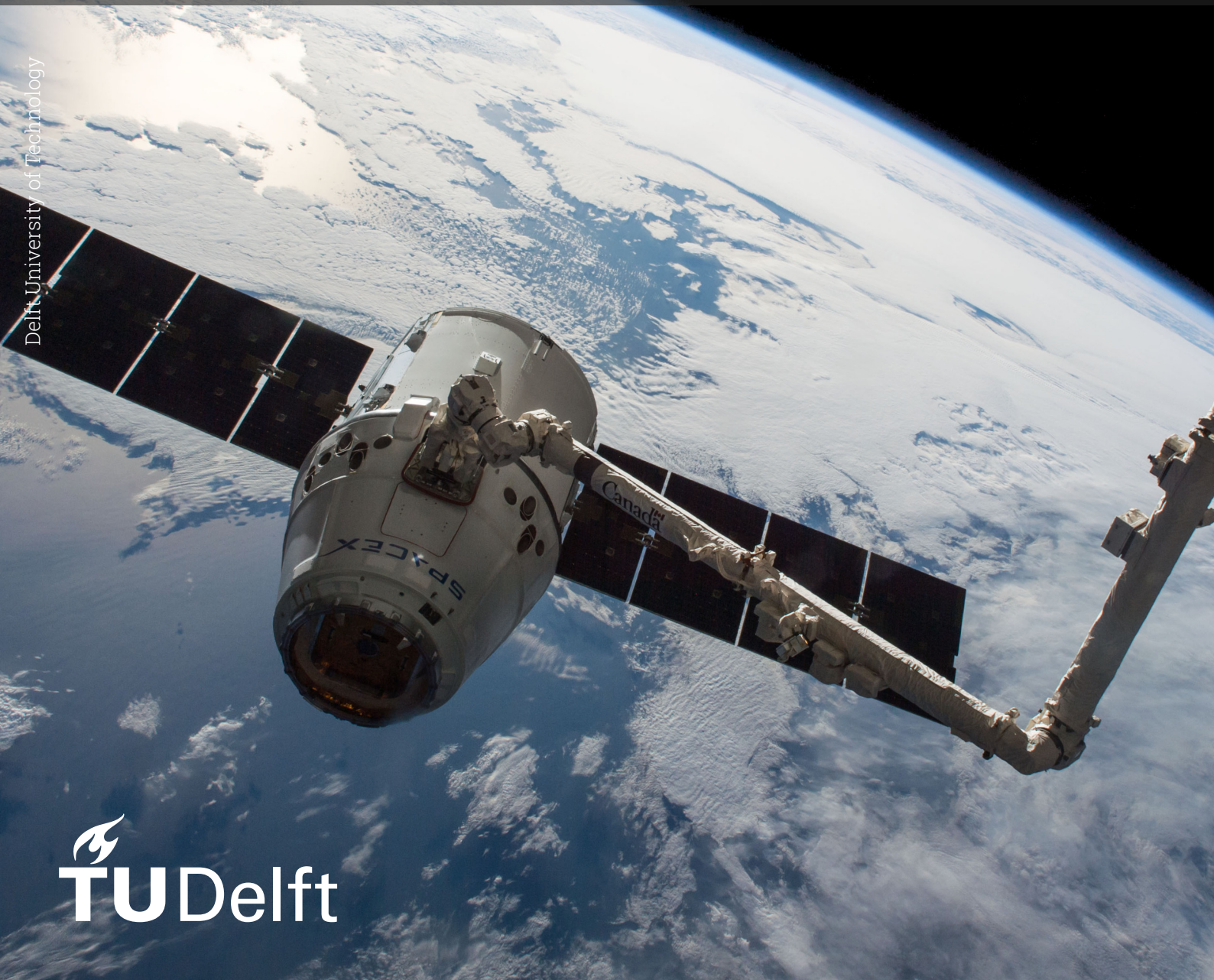
Conditional Diffusion Models for Equity Return Scenario Generation

CS5000: MSc CS Thesis project

Renyi Yang

Delft University of Technology



TU Delft

Conditional Diffusion Models for Equity Return Scenario Generation

by

Renyi Yang

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on TBD

Student number:	5470668	
Project duration:	Janunary 5, 2026 – July 31, 2026	
Thesis committee:	Prof. dr. Lydia Chen,	TU Delft, Chair
	Dr. Shuaiqiang Liu,	TU Delft
	Dr. Daniel Gradeci,	Robeco UK Ltd.
	Ewout Noort	Robeco Nederland B.V.

Cover:	Canadarm 2 Robotic Arm Grapples SpaceX Dragon by NASA under CC BY-NC 2.0 (Modified)
Style:	TU Delft Report Style, with modifications by Daan Zwaneveld

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

A preface...

Renyi Yang
Delft, June 2026

Abstract

Quantitative investment strategies stand or fall on their behaviour in rare and extreme market events, yet the simulators used for stress testing are still dominated by historical bootstrap, which cannot extrapolate beyond observed history, and Gaussian Monte Carlo, which systematically understates fat tails and joint extremes. This thesis investigates whether modern denoising diffusion models can serve as a more faithful scenario factory for equity returns, both unconditionally and under regime-specific conditioning.

We propose a two-stage pipeline for the equity cross-section. A parametric factor model decomposes each stock return into factor exposures plus a idiosyncratic residual via two-stage OLS. Factor returns are then generated by a Diffusion Transformer whose forward process uses anisotropically coloured stable Lévy noise to match the heavy tails and cross-factor co-movement of equity factors. Regime conditioning is realised at inference time as SNR-weighted gradient guidance against a differentiable energy on the clean factor vector.

Evaluated on a 25-year MSCI World daily panel against historical bootstrap and multivariate Gaussian baselines, the proposed model produces factor marginals and joint structure that are closer to held-out data than either baseline, and admits realistic regime-conditioned sampling for tail-stress scenarios. Limitations at the stock-marginal level and a temporal extension are discussed as directions for future work.

Contents

1

Introduction

1.1. Background

In quantitative finance and investment management, particularly within sophisticated strategies employed by hedge funds, the primary concern extends beyond the average or expected performance of a portfolio. Crucially, the behavior of investment strategies during rare and extreme market events, such as the 2008 Global Financial Crisis, the March 2020 COVID-19 shock, liquidity crunches, or sudden volatility spikes, often dictates overall success or failure. Strategies involving leverage, short exposure, and specific factor tilts can exhibit highly non-linear and adverse reactions under market stress. Consequently, the ability to accurately simulate such extreme scenarios is paramount for robust risk management and comprehensive stress-testing of quantitative investment strategies.

Research Gap The core issue with classic generative techniques for financial scenarios is their fundamental limitation in capturing tail behavior. For instance, methods based on historical resampling, such as the stationary bootstrap [26], are inherently backward-looking and struggle to produce novel yet plausible crisis events. Similarly, Monte Carlo simulations that rely on standard assumptions (e.g., Gaussian distributions) often fail to replicate the complex, non-linear co-movements and sudden correlation spikes that characterize real-world financial market stress.

Why Diffusion Models Diffusion models can generate novel and unseen tail scenarios. These models inherently learn complex, time-varying dependencies and non-linear correlations among multiple assets through their iterative denoising process, capturing the intricate co-movement structures that emerge during market distress. Diffusion models provide a flexible framework for both unconditional generation of extreme scenarios and conditional synthesis given specific risk factors.

1.2. Problem Definition

1.2.1. Problem Context

Definition 1 (Scenario). Let $M \geq 1$ denote the number of assets and T the number of time steps. A scenario is return trajectory

$$\mathbf{r} = (r_t)_{t=1}^{T} \in \mathbb{R}^{M \times T},$$

where $r_t \in \mathbb{R}^M$ represents the vector of returns at time t.

Definition 2 (Tail Risk). A trading strategy $\varphi : \{1, \ldots, T\} \times \mathbb{R}^{M \times T} \to \mathbb{R}^M$ maps time and scenario to portfolio weights. For scenario $\mathbf{r}$, the strategy's profit and loss (P&L) is

$$\Pi(\varphi, \mathbf{r}) = \sum_{t=1}^{T} \varphi(t, \mathbf{r})^\top r_t,$$

where $\varphi(t, \mathbf{r}) \in \mathbb{R}^M$ represents holdings at time t given scenario $\mathbf{r}$.

Given a probability distribution P over scenarios and confidence level $\alpha \in (0,1)$, tail risk is quantified by Value-at-Risk:

$$\mathrm{VaR}_\alpha(\varphi, P) := \inf\{x \in \mathbb{R} : P(\Pi(\varphi, \mathbf{r}) \le x) \ge \alpha\}.$$

In this project, we are interested in generating scenarios below a certain threshold of Value-at-Risk.

Definition 3 (Regime).

A regime is a discrete label $\ell \in \mathcal{L}$ that categorizes the market condition of a scenario $\mathbf{r}$. The regime assignment is determined by a mapping $\mathcal{R} : \mathbb{R}^{M\times T} \to \mathcal{L}$ based on statistical indicators such as:

- Volatility: $\sigma_{\text{real}} = \sqrt{\frac{1}{T}\sum_{t=1}^{T} \|r_t\|^2}$
- Maximum drawdown: $\text{MDD} = \max_{1\le s<t\le T} \left(\sum_{i=1}^{s} r_i - \sum_{i=1}^{t} r_i\right)$
- Factor performance metrics, sector/country dominance patterns

Examples: $\mathcal{L} = \{\text{bull}, \text{bear}, \text{crisis}, \text{calm}\}$ or volatility-based bins.

1.2.2. Goals

Goal 1 (Realistic Tail Risk Scenario Sampling). We want to train a diffusion model p_θ from the realistic scenario dataset D_r. Let P_θ be the synthetic histories distribution generated by the diffusion model. For benchmark strategies and typical tail levels, the left-tail P&L risk under P_θ matches that under D_r; evaluate by comparing VaR/ES on held-out data versus synthetic samples, more formally,

$$\sup_{k\in\{1,\dots,K\},\ \alpha\in\mathcal{A}} \left|\mathrm{VaR}_\alpha(\varphi_k, P_\theta) - \mathrm{VaR}_\alpha(\varphi_k, P_r)\right|.$$

In order to model joint extreme event scenarios, we optimize on a subset $\mathcal{I} \subseteq \{1,\dots,K\}$ of multi-asset stress strategies that probe cross-asset co-movements (e.g., baskets, factor spreads), so that joint left-tail risk under P_θ matches that under P_r, enabling effective sampling of collective extremes.

Goal 2 (Regime-Conditioned Scenario Transformation). Given an initial scenario $\mathbf{r}_{\text{orig}}$ and a target extreme regime $\ell \in \mathcal{L}$ (as defined by the mapping $\mathcal{R}$), the goal is to generate a new scenario, $\mathbf{r}_{\text{new}}$. This new scenario should represent a plausible transformation of the original scenario under the conditions of the target regime.

Formally, the task is to sample from the conditional distribution $P(\mathbf{r}_{\text{new}} \mid \mathbf{r}_{\text{orig}}, \ell)$. The generated scenario $\mathbf{r}_{\text{new}}$ must satisfy two primary conditions:

1. **Regime Membership:** It must belong to the target regime, i.e., $\mathcal{R}(\mathbf{r}_{\text{new}}) = \ell$.
2. **Plausible Transformation:** It should preserve key idiosyncratic features of $\mathbf{r}_{\text{orig}}$ that are not directly related to the regime change.

This task is typically achieved by classifier-free guidance [14] with labeled data.

2

Preliminary

2.1. Diffusion Model

Diffusion models represent a powerful class of generative models that transform a simple prior distribution, typically standard Gaussian noise, into a complex data distribution by reversing a gradual noise-injection process. Built upon foundational pillars [36], this development includes Denoising Diffusion Probabilistic Models (DDPMs), Score-Based Generative Models (SGMs), and Score-based Stochastic Differential Equations (Score SDEs).

Definition: Diffusion Model
Let $P_r(\mathbf{x}_0)$ denote the unknown data distribution. A diffusion model is defined by the tuple $\mathcal{M} = (q, p_\theta, T, \beta)$, where T is the number of diffusion steps and $\beta = \{\beta_t\}_{t=1}^T$ is a variance schedule.

1. **Forward Diffusion Process:** A Markov chain that perturbs $\mathbf{x}_0$ by adding Gaussian noise:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{2.1}$$

   The marginal distribution at step t is analytically derived as:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{2.2}$$

   where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

2. **Generative Reverse Process:** A parameterized Markov chain p_θ that approximates the intractable reverse distribution $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad p(\mathbf{x}_T) = \mathcal{N}(0, \mathbf{I}) \tag{2.3}$$

   where $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

The generative capability of these models stems from their ability to reconstruct the original data from noise. While DDPMs [13, 30] focus on predicting the added noise to perform this reversal, SGMs [32, 33] approach the problem by estimating the score function—the gradient of the log-density $\nabla_{\mathbf{x}} \log p(\mathbf{x})$—and using Langevin dynamics for sampling. These perspectives were eventually unified through the framework of Stochastic Differential Equations (SDEs) [34], where the forward and reverse processes are viewed as continuous-time evolutions. This unification allows for deterministic sampling through "Probability Flow ODEs," enabling exact likelihood computation and more efficient numerical solvers [17].

Over time, the architectural backbone of these models has evolved from basic multi-layer perceptrons to sophisticated U-Nets equipped with self-attention and group normalization. More recently, Transformer-based architectures like Diffusion Transformers (DiT) and specialized structures for temporal data have gained prominence.

In the context of financial scenario generation, these models are particularly valuable because their iterative denoising process inherently learns the time-varying dependencies and non-linear correlations that characterize market distress, providing a flexible framework for both unconditional and conditional synthesis of extreme market regimes [36].

In order to perform our task, we focus on two key features of diffusion models: conditional sampling and temporal modeling. Conditional sampling allows us to generate scenarios that adhere to specific market regimes. Temporal modeling is curcial for capturing the volatility clustering and co-movement patterns especially during extereme market conditions. 2.1.1 and 2.1.2 provide a detailed review of these two aspects, respectively.

2.1.1. Conditional Sampling

Conditional sampling in diffusion models aims to generate samples $\mathbf{x}_0$ that satisfy specific external constraints or conditions $\mathbf{c}$, effectively modeling the posterior distribution $p(\mathbf{x}_0|\mathbf{c})$. In the domain of financial modeling, these conditions might represent specific market regimes, factor exposures, or macroeconomic indicators. Conditional sampling strategies can be broadly categorized into training-time (built-in) conditioning and inference-time (post-hoc) conditioning.

Training-time Conditioning

Built-in conditioning involves the joint learning of the data distribution and the condition during the training phase. In this approach, the denoiser $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{c})$ is explicitly dependent on the condition $\mathbf{c}$, allowing the model to learn $p(\mathbf{x}|\mathbf{c})$ directly. Achieving this typically requires architectural modifications that inject conditional information into the network. Discrete or continuous conditions are first mapped through learned embedding layers and then either concatenated with or added to the timestep embeddings; these augmented representations are injected into each residual block of the U-Net or Transformer architecture [24]. Another popular mechanism is adaptive normalization, where techniques such as Adaptive Group Normalization or Adaptive Layer Normalization scale and shift the normalized feature maps using parameters predicted from the condition, providing a more expressive way to modulate the generative process [7]. Finally, classifier-free guidance (CFG) offers a streamlined training scheme by jointly optimizing conditional and unconditional objectives—using a null token to represent the absence of a condition—and then combining the two predictions at inference time through an extrapolation factor; this improves sample quality and alignment at the cost of reduced diversity [14].

Inference-time Conditioning

Post-hoc conditioning allows constraints to be imposed on a pre-trained unconditional diffusion model $p(\mathbf{x})$ during the sampling process, obviating the need for retraining and enabling arbitrary guidance modalities. Methods in this category typically fall into two groups, soft and hard guidance. Soft guidance techniques add a penalty or guiding gradient at each reverse diffusion step to nudge the sample trajectory toward regions of high conditional probability. For example, classifier guidance trains a separate model $p(\mathbf{c}|\mathbf{x}_t)$ on the noisy data and uses the gradient $\nabla_{\mathbf{x}_t} \log p(\mathbf{c}|\mathbf{x}_t)$ to adjust the mean of the reverse step [7]. This idea can be generalized by employing the gradient of an arbitrary differentiable loss $\mathcal{L}(\mathbf{x}_t, \mathbf{c})$ or an energy-based parameterization, as proposed in recent work on loss-based and energy-based guidance [31]. Another extension, diffusion posterior sampling (DPS), makes use of manifold-constrained gradients to handle complex observation models, which is especially useful for inverse problem settings [5]. In contrast, hard guidance methods directly modify the intermediate latents $\mathbf{x}_t$ to enforce exact constraints. A straightforward tactic is replacement and masking (as in RePaint), where known or fixed regions—such as parts of an image in inpainting tasks—are substituted with their forward-diffused counterparts at each reverse step, compelling the model to fill in the remaining regions consistently [21]. Structural guidance techniques like Iterative Latent Variable Refinement (ILVR) and SDEdit further constrain generation by matching low-frequency components to a reference signal or by perturbing a guide image with noise and denoising it so that the overall structure is preserved while realistic detail is added [4, 23].

2.1.2. Temporal Modeling

Temporal modeling is a critical dimension of diffusion-based generation, particularly when dealing with time series where sequential dependencies and temporal coherence are paramount. The integration of temporal structures into the diffusion framework allows for the capture of complex dynamics that

traditional autoregressive models often struggle to represent. In financial applications, this capability is essential for modeling phenomena such as volatility clustering, mean reversion, and the co-movement of multiple assets across different time scales.

The architectural evolution of temporal diffusion models has been marked by a transition from standard U-Nets to more specialized backbones designed to handle sequential data. One of the pioneering works in this area is TimeGrad [27], which employs an autoregressive recurrent neural network (RNN) to model the temporal evolution of the data distribution, with a diffusion model serving as the conditional generator at each time step. By predicting the gradient of the log-density of the next observation given the historical context, TimeGrad effectively captures the multi-modal nature of future trajectories in high-dimensional financial time series.

Beyond autoregressive approaches, conditional score-based models have also shown significant promise for structured temporal tasks. CSDI (Conditional Score-based Diffusion models for probabilistic time series Imputation) [35] represents a major advancement in this regard, utilizing a 2D-attention mechanism that simultaneously considers both the temporal dependencies and the correlations between different features or variables. This dual-attention structure allows the model to leverage information from observed values across both the time and feature dimensions to reconstruct missing data or generate future scenarios with high fidelity.

More recently, the adoption of Structured State Space Models (SSMs) and Transformer-based architectures has further enhanced the efficiency and expressive power of temporal diffusion models. SSSD (Structured State Space Diffusion) [1] replaces the traditional dilated convolutions or RNNs with state-space layers, which are particularly effective at capturing long-range dependencies while maintaining computational efficiency. Additionally, models like DiffWave [18] and its derivatives have demonstrated the utility of bidirectional temporal information and multi-scale feature extraction in generating coherent high-frequency signals. These developments collectively provide a robust toolkit for financial scenario generation, enabling the synthesis of realistic market paths that respect both short-term fluctuations and long-term structural trends.

2.2. Scenario Generation

Scenario generation is a foundational component of modern financial risk management, providing the necessary inputs for stress testing, capital allocation, and counterfactual analysis. In the context of risk management, the core objective is to synthesize a diverse set of alternative realizations—or "scenarios"—that accurately reflect the underlying joint distribution and temporal dependencies of market risk factors. By generating counterfactual versions of market conditions, these frameworks enable risk managers to probe the structural vulnerabilities of complex portfolios and evaluate the stability of financial systems under adverse or tail-risk regimes that are often underrepresented in historical datasets. The downstream applications of generated scenarios—including risk management and portfolio optimization—are detailed in Appendix A.2.

2.2.1. Traditional Methods

Traditional scenario generation methods are primarily divided into resampling-based techniques and parametric simulation models. Among the most influential is the Stationary Bootstrap [26], which samples blocks of random lengths following a geometric distribution to preserve the local temporal dependencies of market returns.

To bridge the gap between historical resampling and time-varying volatility, Filtered Historical Simulation (FHS) was proposed in [2]. By scaling historical residuals with current GARCH-based volatility estimates, FHS allows for the generation of scenarios that respect current market regimes while retaining the non-parametric nature of return innovation distributions.

For multi-asset modeling, Copula-based simulations are a standard industry tool for capturing non-linear dependencies. By decoupling marginal distributions from the dependence structure, these models [9] can effectively represent tail dependence and systemic risk, which are often overlooked by traditional linear correlation measures. Despite their utility, these approaches often struggle to capture the high-dimensional complexities of global portfolios compared to modern machine-learning frameworks.

2.2.2. Generative AI Methods

The emergence of deep generative models has fundamentally shifted the frontier of scenario generation, enabling data-driven synthesis that goes beyond the parametric and resampling limitations of traditional approaches. Rather than relying on fixed distributional assumptions, these models learn the full joint distribution of asset returns from data, capturing complex non-linear dependencies and tail behaviour that classical methods cannot represent.

Early work in this direction applied VAE-based architectures to factor models. FactorVAE [8] proposes a probabilistic dynamic factor model that integrates the VAE framework with the dynamic factor model (DFM), treating latent factors as random variables to explicitly model noise. A GRU-based feature extractor encodes historical stock characteristics, and a prior-posterior learning scheme guides factor extraction. MacroVAE [19] instead focuses on conditioning generation on macroeconomic context. Using a convolutional ResNet encoder-decoder with Feature-wise Linear Modulation (FiLM), it generates 22-day multivariate return sequences conditioned on macro indicators, enabling counterfactual scenario analysis under macroeconomic conditions not observed historically.

More recently, diffusion models have been applied to cross-sectional return generation. FactorDiff [12] adapts the Diffusion Transformer (DiT) with token-wise conditioning, treating each asset as a token conditioned on its own observable factor vector via AdaLN-Zero modulation. Cross-asset dependencies are captured through multi-head self-attention. FactorDM [3] takes a different approach: rather than conditioning on observable factors, it proposes an unconditional score-based diffusion model that embeds latent factor structure directly into the score function via time-varying orthogonal projections, decomposing the score into a low-dimensional subspace component and an idiosyncratic complement.

Table 2.1 summarises these four models across key design dimensions.

Table 2.1: Comparative analysis of generative AI models for financial scenario generation. N: assets; T: time steps; C: characteristics per asset; K: factors.

Model	Framework	Backbone	Output Shape	Condition c	Injection	Temporal
FactorVAE [8]	Probabilistic VAE	GRU + MLP	$(N,)$	Hist. characteristics $(N\times T\times C)$	GRU encoder → factor posterior	N
MacroVAE [19]	Conditional VAE	Conv ResNet2D + FiLM	$(T\times N)$	Macro indicators $c \in \mathbb{R}^{36}$	FiLM modulation in decoder	Y
FactorDiff [12]	Conditional DDPM	DiT (token-wise)	$(N,)$	Factor matrix $X_t \in \mathbb{R}^{N\times K}$	Token-wise AdaLN-Zero per asset	N
FactorDM [3]	Score-based DM	Score-decomposed MLP	$(N,)$	None (unconditional)	Latent factor in score decomp.	N

3

Methodology

Our proposed framework synthesises equity returns by combining two components. A parametric factor model that converts between stock returns and factor returns via fixed factor loadings and a Diffusion Transformer (DiT) that models the joint distribution of factor returns. Scenario generation starts from sampling factor returns from the DiT, and reconstruction back to stock returns given fixed factor loadings and a sample from per-stock idiosyncratic noise model.

The factor model is estimated once via ordinary least squares and treated as fixed throughout. The diffusion model is the only learned generative object, it captures the cross-factor co-movement, the heavy-tailed marginals of factor returns and other stylistic features of the data. The remainder of this chapter is organised as follows. Section 3.1 formalises the parametric factor model and details how loadings are estimated. Section 3.2 describes the factor diffusion model: the heavy-tailed forward process, the Gaussian-parameterised reverse process, and the Transformer-based denoiser with its training objective. Section 3.3 extends the model to temporally coherent paths by conditioning the denoiser on the previous day's cross-section and unrolling a first-order Markov chain. Section 3.4 ties the components together as algorithm boxes for training and sampling.

3.1. Parametric Factor Decomposition

The parametric factor model decomposes each stock return into a deterministic part driven by a small number of generric factors and a stochastic idiosyncratic part specific to each stock. Once the factor loadings are estimated, the only object that requires generative modelling is the joint distribution of factor returns, which lives in a much lower-dimensional space than the stock return vector itself.

The Capital Asset Pricing Model [29] states that the expected excess return of stock i is proportional to the expected excess return of the market portfolio:

$$\mathbb{E}_t[R_{i,t+1}] - R_t^f = \beta_{i,t}\left(\mathbb{E}_t[R_{t+1}^m] - R_t^f\right). \tag{3.1}$$

The CAPM is an economic model in expectations; turning it into an estimable form requires introducing a mean-zero residual $\varepsilon_{i,t+1}$ and treating β as constant within an estimation window. It has been long established that a single market factor does not fully explain the cross-section of expected returns [10], motivating the extension to a vector of K factors. The empirical multi-factor model is

$$R_{i,t+1} - R_t^f \;=\; a_i \;+\; \boldsymbol{\beta}_i^\top \mathbf{F}_{t+1} \;+\; \varepsilon_{i,t+1}, \tag{3.2}$$

where $\mathbf{F}_{t+1} \in \mathbb{R}^K$ is the realised factor return vector, $\boldsymbol{\beta}_i \in \mathbb{R}^K$ collects the per-stock loadings, a_i is a stock-specific intercept, and $\varepsilon_{i,t+1}$ captures the residual unexplained by the factors. Our proposed framework treats $\boldsymbol{\beta}_i$ and a_i as deterministic and estimated once on the training set, while the joint distribution of $\mathbf{F}$ and the per-stock distribution of ε_i are the only stochastic objects that subsequent components model.

We estimate all parts of Equation (3.2) in two sequential ordinary-least-squares (OLS) stages, following the cross-sectional regression methodology of Fama and MacBeth [11].

First stage is to do cross-sectional regression to get factor returns. At each date t, the contemporaneous stock returns are regressed on the lagged firm characteristics $B_{t-1} \in \mathbb{R}^{N\times K}$:

$$R_{i,t} \;=\; \mathbf{b}_{i,t-1}^{\top}\mathbf{F}_t \;+\; u_{i,t}, \qquad i = 1,\dots,N, \tag{3.3}$$

yielding a single realisation $\mathbf{F}_t$ of the factor return vector for that date. Repeating this regression at every date produces the historical factor return series $\{\mathbf{F}_t\}_{t=1}^{T}$, the data on which the diffusion model is later trained. We treat the intercept $u_{i,t}$ as the market factor. Given the factor return series, the second stage is to calculate the loadings of each individual stock via per-stock time-series OLS according to Equation (3.2).

Once the factors and factor loadings are frozen, the residual series $\{\varepsilon_{i,t}\}_{t=1}^{T}$ of each stock is treated as samples from a stationary, stock-specific distribution. Rather than assuming Gaussian residuals, which systematically underestimate equity tail risk [6], we fit a scaled Student-t distribution per stock to the standardised residuals:

$$\varepsilon_{i,t}/\sigma_i \;\sim\; t_{\nu_i}(0,1), \tag{3.4}$$

where $\sigma_i = \sqrt{\mathbb{E}[\varepsilon_{i,t}^2]}$ is the sample residual standard deviation and ν_i is the degrees-of-freedom parameter, estimated by maximum likelihood according to the empirical distribution.

At sampling time, idiosyncratic noise is drawn as

$$\varepsilon_{i,s} \;=\; \sigma_i \sqrt{\frac{\nu_i - 2}{\nu_i}} \cdot z_{i,s}, \qquad z_{i,s} \sim t_{\nu_i}(0,1), \tag{3.5}$$

where the rescaling factor $\sqrt{(\nu_i - 2)/\nu_i}$ corrects for the inflated variance of the distribution.

Modelling factor returns instead of stock returns directly carries several practical advantages. First learning the joint distribution of factors rather than the full stock returns reduces the generative problem by more than two orders of magnitude. Then the decomposition is also robust to stock-level missingness that would severely undermine a model trained on all stocks jointly. In our proposed method, missing observations only affect the factor loading estimation for the affected stock, while the factor return series and the diffusion model itself remain unaffected.

Beyond these data-handling benefits, the decomposition also makes downstream usage cleaner. Conditioning the diffusion model on factor-level signals(for example, a low-momentum stress regime) is both economically meaningful and statistically efficient, because each factor aggregates information across the entire cross-section rather than being tied to a single stock. Likewise, stocks outside the training universe can be incorporated at sampling time without retraining the diffusion model. This is structurally impossible for a generative model that operates directly on the joint stock return distribution.

3.2. Factor Diffusion Model

The factor diffusion model contributes to the generative component in the proposed framework. It captures the joint distribution of factor returns $\mathbf{F}_t$, from which all stock-level scenarios are subsequently reconstructed via the projection introduced in Section 3.1. The model follows the denoising-diffusion paradigm reviewed in Section 2.1. Two design choices distinguish our implementation from a standard image-domain DDPM, each motivated by the structure of financial factor returns. First, the forward process replaces Gaussian noise with α-stable (Lévy) noise, anisotropically coloured by the empirical factor correlation matrix, so that the noising trajectory matches the fat-tailed marginal and cross-sectional dependence of the data. Second, the denoiser is a stack of Transformer blocks rather than a convolutional U-Net: multi-head self-attention naturally captures cross-factor co-movement, and AdaLN provides an efficient mechanism for injecting both the diffusion timestep and auxiliary conditioning signals.

Figure 3.1 summarises the resulting architecture. The remainder of this section describes each component in detail: the forward process (Section 3.2.1), the reverse process and conditional sampling (Section 3.2.2), and the denoiser together with its training objective (Section 3.2.3).

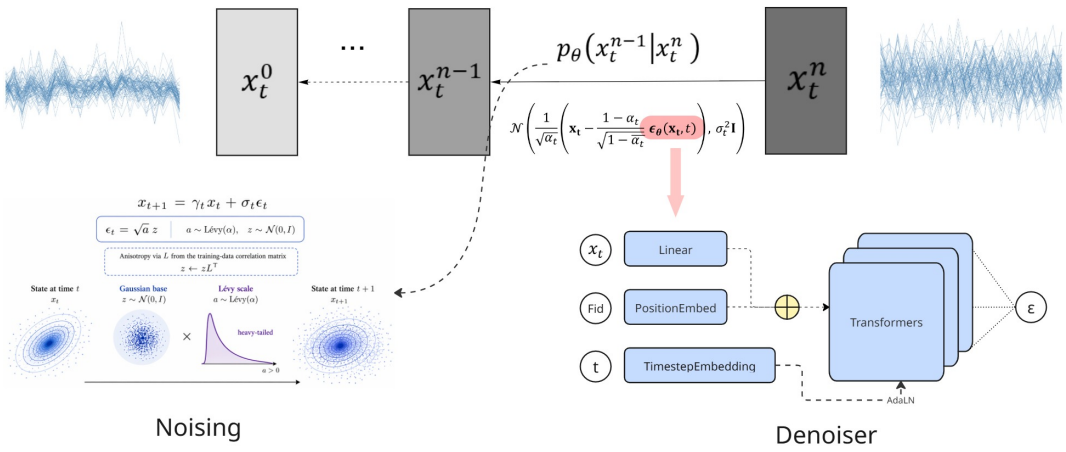


Figure 3.1: Architecture Overview of the factor diffusion model. The forward process (left) combines a Gaussian base with a Lévy scale to produce heavy-tailed perturbations of the factor return vector x_t, anisotropically coloured by the empirical factor correlation matrix. The denoiser (right) embeds the noised input x_t together with a per-factor positional embedding and the diffusion timestep t; these embeddings feed a stack of Transformer blocks whose layer normalisations are modulated by the timestep through AdaLN, and whose output predicts the noise component $\epsilon_\theta(x_t, t)$.

3.2.1. Forward Process: Heavy-Tailed Noising

The forward noising process follows the standard DDPM [13] with two modifications: the Gaussian increment is replaced by heavy-tailed alpha stable noise inspired by the Denoising Lévy Probabilistic Model (DLPM, [28]), and the Gaussian component of that noise is anisotropically coloured by the empirical factor correlation matrix following the blue-noise diffusion construction of Huang et al. [15].

Heavy-tailed α-stable forward chain. Following DLPM, we replace the Gaussian increment by a draw from the symmetric α-stable distribution $S(\alpha, 0)$, which recovers the standard normal at $\alpha = 2$ and is heavy-tailed for $\alpha < 2$:

$$\mathbf{x}_t \;=\; \gamma_t\, \mathbf{x}_{t-1} \;+\; \sigma_t\, \boldsymbol{\varepsilon}_t, \qquad \boldsymbol{\varepsilon}_t \;\sim\; S(\alpha, 0). \tag{3.6}$$

The schedule coefficients are constrained to preserve the α-norm scale of the marginal, $\gamma_t^\alpha + \sigma_t^\alpha = 1$, generalising the variance-preserving DDPM constraint $\alpha_t + \beta_t = 1$. We retain a cosine $\bar{\alpha}_t$ schedule [24] and define $\gamma_t = \alpha_t^{1/\alpha}$, $\sigma_t = \beta_t^{1/\alpha}$, with cumulative coefficients $\bar{\gamma}_t = \prod_{s \le t} \gamma_s$ and $\bar{\sigma}_t = (1 - \bar{\gamma}_t^\alpha)^{1/\alpha}$. The closed-form marginal is

$$\mathbf{x}_t \;=\; \bar{\gamma}_t\, \mathbf{x}_0 \;+\; \bar{\sigma}_t\, \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim S(\alpha, 0). \tag{3.7}$$

At $\alpha = 2$, equations (3.6)–(3.7) reduce exactly to DDPM. For $\alpha < 2$ the increment $\boldsymbol{\varepsilon}_t$ has infinite variance, which precludes a direct MSE training objective. Following DLPM, we use the Gaussian scale-mixture representation of $S(\alpha, 0)$,

$$\boldsymbol{\varepsilon}_t \;=\; \sqrt{A_t}\, \mathbf{z}_t, \qquad A_t \sim S\!\left(\tfrac{\alpha}{2}, 1\right), \quad \mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{3.8}$$

where the positive subordinator A_t carries the heavy-tailed scale and is sampled independently per factor. Conditional on the realised subordinator path, the marginal becomes parameterized Gaussian:

$$\mathbf{x}_t \mid \mathbf{x}_0,\, A_{1:t} \;\sim\; \mathcal{N}(\bar{\gamma}_t\, \mathbf{x}_0,\; \Sigma_t), \quad \Sigma_t = \sigma_t^2 A_t + \gamma_t^2 \Sigma_{t-1}. \tag{3.9}$$

Anisotropic colouring Huang et al. [15] showed for image diffusion that replacing such isotropic Gaussian noise with a correlated Gaussian, obtained by multiplying a white draw with a fixed Cholesky factor $\mathbf{L}$ of a target covariance, can substantially improve generation quality; the principle is the standard reparameterisation $\mathbf{Lz} \sim \mathcal{N}(\mathbf{0}, \mathbf{L}\mathbf{L}^\top)$. We instantiate the same construction for factor returns by taking

$\mathbf{L}$ as the Cholesky factor of the empirical factor correlation matrix $\mathbf{C} = \mathbf{L}\mathbf{L}^\top$, computed once on the standardised training set, and colouring the Gaussian component of the scale mixture:

$$\boldsymbol{\varepsilon}_t \;=\; \sqrt{A_t}\,\mathbf{L}\,\mathbf{z}_t. \tag{3.10}$$

The subordinator A_t is still sampled independently per factor, so the heavy-tailed component remains anisotropic in the same sense as the data, while the Gaussian core respects the empirical co-movement structure. Setting $\mathbf{C} = \mathbf{I}$ recovers the original DLPM formulation; setting additionally $\alpha = 2$ recovers DDPM.

Stable training via Monte Carlo on the subordinator. In practice, the subordinator distribution $S(\alpha/2, 1)$ is itself heavy-tailed and a small fraction of draws produce values two-to-three orders of magnitude above the bulk even after the safety clip; a single such draw in a minibatch yields a loss spike that dominates the gradient and pulls the optimiser toward a degenerate constant-output solution. We therefore estimate the per-sample loss with a median-of-means scheme over independent subordinator draws [22]. at each training step we draw M outer subordinators $A^{(1)}, \ldots, A^{(M)}$ and, conditional on each, K_z inner Gaussian draws; the squared errors are averaged over the K_z inner draws and the median is taken across the M outer draws as the per-sample loss. The inner average is a Rao–Blackwell-style variance reduction on the Gaussian factor of the mixture; the outer median provides robustness to the rare extreme A.

3.2.2. Reverse Process: (Conditional) Parameterised Denoising

The forward chain has a heavy-tailed marginal but is conditionally Gaussian given the subordinator path $A_{1:T}$. We exploit this structure by parameterising the reverse process as a Gaussian denoiser conditioned on $A_{1:T}$, training it with the standard noise-prediction objective, and integrating out the subordinator at sampling time via Monte Carlo.

Conditional on $A_{1:t}$, the forward marginal in equation (3.9) is Gaussian, so the reverse posterior $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0, A_{1:t})$ is also Gaussian and admits a closed form. Mirroring the DDPM derivation [13] but with the heavy-tailed scaling, one obtains

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0, A_{1:t}) \;=\; \mathcal{N}\big(\boldsymbol{\mu}_t,\; \Gamma_t\,\Sigma_{t-1}\,\mathbf{I}\big)\,, \tag{3.11}$$

with posterior contraction factor and mean

$$\Gamma_t \;=\; 1 - \frac{\gamma_t^2\,\Sigma_{t-1}}{\Sigma_t}, \qquad \boldsymbol{\mu}_t \;=\; \frac{\mathbf{x}_t - \bar{\sigma}_t\,\Gamma_t\,\boldsymbol{\varepsilon}}{\gamma_t}. \tag{3.12}$$

We follow DDPM [13] and replace the unknown noise $\boldsymbol{\varepsilon}$ in (3.12) with a learned estimate $\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t)$. The parameterised reverse step is

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, A_{1:t}) \;=\; \mathcal{N}\!\left(\frac{\mathbf{x}_t - \bar{\sigma}_t\,\Gamma_t\,\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t)}{\gamma_t},\; \Gamma_t\,\Sigma_{t-1}\,\mathbf{I}\right). \tag{3.13}$$

The denoiser network does not need to model the subordinator: $A_{1:t}$ enters only through the schedule scalars $\gamma_t, \bar{\sigma}_t, \Sigma_t, \Gamma_t$. The training objective therefore reduces to a noise-prediction MSE, with the heavy-tailed structure folded entirely into the schedule and the subordinator sampling at inference time.

At sampling time we draw the subordinator path $A_{1:T}$ once per trajectory, accumulate $\Sigma_t = \sigma_t^2 A_t + \gamma_t^2 \Sigma_{t-1}$, initialise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \Sigma_T \mathbf{I})$ jointly with the same chain, and unroll equation (3.13) from $t = T$ down to $t = 1$.

In order to sample from a conditional distribution that biases trajectories toward a target regime $\mathbf{c}$. We encode such regimes as a differentiable loss function $\mathcal{L}(\mathbf{x}_0, \mathbf{c})$ on the clean factor return vector. Our modeling target becomes:

$$p^*(\mathbf{x}_0 \mid \mathbf{c}) \;\propto\; p_\theta(\mathbf{x}_0)\,\exp\big(-s\,\mathcal{L}(\mathbf{x}_0, \mathbf{c})\big), \tag{3.14}$$

where $s > 0$ controls the strength of the bias. We approximate samples from p^* using the energy-based diffusion guidance scheme reviewed in Section 2.1.1 [31, 7]. At each reverse step we estimate $\mathbf{x}_0$ from the current $\mathbf{x}_t$ via the Tweedie identity adapted to the heavy-tailed schedule,

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) \;=\; \frac{\mathbf{x}_t - \bar{\sigma}_t\, \hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t)}{\bar{\gamma}_t}, \tag{3.15}$$

and shift the posterior mean of equation (3.12) along the energy gradient,

$$\boldsymbol{\mu}_t \;\leftarrow\; \boldsymbol{\mu}_t \;-\; s\, w_t\, \Gamma_t \Sigma_{t-1}\, \nabla_{\mathbf{x}_t} \mathcal{L}(\hat{\mathbf{x}}_0(\mathbf{x}_t), \mathbf{c})\,. \tag{3.16}$$

Because the Tweedie estimate $\hat{\mathbf{x}}_0$ is less unreliable at large timesteps We set

$$w_t \;=\; \left(\frac{\mathrm{SNR}_t}{\mathrm{SNR}_t + 1}\right)^p, \qquad \mathrm{SNR}_t = \frac{\bar{\gamma}_t^2}{\bar{\sigma}_t^2}, \tag{3.17}$$

which damps the guidance update during the high-noise stage of the reverse chain. The factor $w_t \in [0, 1]$ is inspried from Bayes-optimal denoiser weight in the EDM family [17] : $w_t \to 0$ as $t \to T$ and $w_t \to 1$ as $t \to 0$. The decay exponent p controls the sharpness of the transition; we use $p = 1$ throughout, w_t act as a penalty coefficient for variable sampling timestamps that interpolates smoothly between the high-noise and low-noise regimes.

The exact target distribution p^* in (3.14) can also be sampled by soft rejection on top of the unconditional model: draw $\mathbf{x}_0 \sim p_\theta$ and accept with probability $\exp(-s\,\mathcal{L}(\mathbf{x}_0, \mathbf{c}))$. Soft rejection preserves the unconditional joint distribution exactly on the unconstrained coordinates and is the gold standard whenever the acceptance rate is tractable; gradient guidance is an inference-time approximation that becomes exact in the limit $s \to 0$ and small reverse step size [31]. We use guidance as the production sampler because of its lower compute cost.

3.2.3. Denoiser

The denoiser $\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t)$ is a Transformer encoder operating on a sequence of factor tokens, mirroring the Diffusion Transformer (DiT) backbone [25]. We adopt this design over the convolutional U-Net used in image diffusion because multi-head self-attention treats every factor pair symmetrically and is well suited to capturing the dense cross-factor co-movement that the model is required to reproduce.

The noised input $\mathbf{x}_t \in \mathbb{R}^K$ is split into K scalar coordinates and lifted to a d-dimensional embedding by a per-coordinate linear projection. A learnable per-factor identity embedding $\mathbf{e}_k \in \mathbb{R}^d$ is added so that each token carries its factor index through the attention layers, playing the role of the positional encoding in a standard Transformer but indexing the factor identity rather than position in time:

$$\mathbf{h}_k^{(0)} \;=\; \mathbf{W}_{\text{in}}\, x_{t,k} \;+\; \mathbf{e}_k, \qquad k = 1, \ldots, K. \tag{3.18}$$

Treating each factor as its own token, rather than concatenating the entire factor vector into a single token, preserves per-factor structure throughout the network and lets attention act directly on factor pairs.

The diffusion timestep t is encoded with a sinusoidal embedding followed by a small MLP, producing a continuous conditioning vector $\mathbf{c}_t$. We inject $\mathbf{c}_t$ into every Transformer block via adaptive layer normalisation (AdaLN), in which the affine parameters of LayerNorm are predicted from $\mathbf{c}_t$ rather than learned as free parameters [25, 7]. This concentrates the timestep dependence into a low-cost modulation while leaving the attention weights themselves unconditioned on t. The body of the denoiser is then a stack of such Transformer blocks; the final hidden states are projected back to scalars by a linear output head, yielding the predicted noise $\hat{\varepsilon}_{\theta,k}(\mathbf{x}_t, t)$ for each factor k.

The denoiser is trained with the standard noise-prediction MSE of [13], adapted to the heavy-tailed forward chain via the Gaussian scale-mixture representation of equation (3.8). For a sample $\mathbf{x}_0$ from the empirical factor return distribution and a uniformly drawn timestep $t \sim \mathcal{U}\{1, \ldots, T\}$, the per-sample loss is

$$\ell(\theta;\, \mathbf{x}_0, t) \;=\; \mathbb{E}_{A,\mathbf{z}}\left[\, \left\| \hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t) - \sqrt{A}\,\mathbf{L}\mathbf{z} \right\|_2^2 \right], \qquad \mathbf{x}_t = \bar{\gamma}_t\, \mathbf{x}_0 + \bar{\sigma}_t\, \sqrt{A}\,\mathbf{L}\mathbf{z}, \tag{3.19}$$

with $A \sim S(\alpha/2, 1)$ sampled per factor and $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The expectation is estimated with the median-of-means scheme over subordinator draws.

3.3. Temporal Factor Generation

The model of Section 3.2 draws a single cross-section $\mathbf{F}$ from the unconditional factor distribution, with no notion of time. Most downstream uses, however, multi-day risk, path-dependent stress and trajectory-level portfolio evaluation, require sequences of cross-sections that are temporally coherent rather than independent daily draws. We obtain such sequences by making the diffusion transformer *autoregressive*: the denoiser is conditioned on the previous day's factor cross-section, and a multi-day path is produced by unrolling a first-order Markov chain over calendar days. We index calendar days by d and write $\mathbf{F}_d \in \mathbb{R}^K$ for the factor cross-section on day d and H for the horizon; the diffusion timestep t and the noised state $\mathbf{x}_t$ retain their meaning from Section 3.2. We make the first-order Markov assumption

$$\mathbf{F}_d \;\perp\; \mathbf{F}_{<d-1} \;\mid\; \mathbf{F}_{d-1}, \tag{3.20}$$

so that the joint law of a path factorises as $p(\mathbf{F}_0, \ldots, \mathbf{F}_{H-1}) = p(\mathbf{F}_0) \prod_{d=1}^{H-1} p(\mathbf{F}_d \mid \mathbf{F}_{d-1})$. Only the denoiser's conditioning input changes; the heavy-tailed forward process, the schedule and the parameterised reverse step of Section 3.2 are reused unchanged.

We augment the denoiser with the previous day's cross-section as an additional conditioning signal, written $\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t, \mathbf{F}_{d-1})$. Following the DiT design [25], this signal enters through exactly the AdaLN pathway already used for the diffusion timestep in Section 3.2.3. The full previous cross-section is mapped to a context vector by a small MLP and combined with the per-factor identity embedding and the timestep context, so that the per-token condition of equation (3.18) becomes

$$\mathbf{g}_k \;=\; \underbrace{\mathrm{MLP}_{\mathrm{cond}}(\mathbf{F}_{d-1})}_{\text{previous-day context}} \;+\; \mathbf{e}_k^{\mathrm{cond}} \;+\; \mathbf{c}_t^{\mathrm{time}}, \qquad k = 1, \ldots, K, \tag{3.21}$$

where $\mathbf{c}_t^{\mathrm{time}}$ is the sinusoidal timestep embedding of Section 3.2.3 and $\mathbf{e}_k^{\mathrm{cond}}$ is a learnable per-factor conditioning identity. Encoding the *entire* previous cross-section, rather than each factor's own lag in isolation, lets every factor's next-day dynamics depend on the whole previous regime, so that cross-factor spillover is carried through time as well as within a day. The denoiser thereby learns the one-step transition density $p_\theta(\mathbf{F}_d \mid \mathbf{F}_{d-1})$.

This temporal condition is orthogonal to the regime condition $\mathbf{c}$ of Section 3.2.2: $\mathbf{F}_{d-1}$ enters the network as an *input* that shapes the learned transition, whereas $\mathbf{c}$ acts at inference time through the energy-gradient shift of equation (3.16). A single reverse step can therefore carry both, advancing the chain by one day while simultaneously biasing that day toward a target regime.

A purely transition-based model cannot start a path on its own, since the first day $\mathbf{F}_0$ has no predecessor. We resolve this with a learned *null* (zero) condition $\varnothing$, a single learnable vector that replaces $\mathrm{MLP}_{\mathrm{cond}}(\mathbf{F}_{d-1})$ in equation (3.21) and represents the absence of a previous day. To train both behaviours into one set of weights we use classifier-free-style condition dropout [14]: for each training pair the previous-day context is replaced by $\varnothing$ with probability p_{drop}. The denoiser then jointly learns the transition $p_\theta(\mathbf{F}_d \mid \mathbf{F}_{d-1})$ and the initial marginal

$$p_\theta(\mathbf{F}_0 \mid \varnothing) \;=\; p_\theta(\mathbf{F}_0), \tag{3.22}$$

which is exactly the unconditional factor distribution of Section 3.2. A path can thus be *self-started* from $\varnothing$ with no external seed, while the same weights still propagate any given day forward.

Generation unrolls the Markov chain of equation (3.20). We first draw $\mathbf{F}_0 \sim p_\theta(\cdot \mid \varnothing)$ from the null condition, then iterate

$$\mathbf{F}_d \;\sim\; p_\theta(\cdot \mid \mathbf{F}_{d-1}), \qquad d = 1, \ldots, H-1, \tag{3.23}$$

feeding each freshly generated cross-section back as the condition for the next. Each draw in (3.23) is a complete reverse-diffusion pass over $t = T, \ldots, 1$, identical to that of Section 3.2.2 except that the denoiser is queried as $\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t, \mathbf{F}_{d-1})$; the heavy-tailed subordinator, schedule scalars and posterior step are unchanged. When a real history is available the rollout can instead be seeded with an out-of-sample day as $\mathbf{F}_0$, after which the chain proceeds identically. Because the temporal and regime conditions

are orthogonal, the guidance update of equation (3.16) can be applied at every step to bias the entire trajectory toward a target regime.

Algorithm 3, stated after the pipeline in Section 3.4, summarises the resulting training and sampling changes relative to Algorithms 1 and 2; all unchanged components, the forward noising internals, the median-of-means loss estimator and the stock-level reconstruction, are reused verbatim and omitted there.

3.4. Generation Pipeline

The components developed in Sections 3.1 –3.2 compose into a two-stage pipeline. Algorithm 1 produces the fixed factor model, the diffusion schedule, the empirical correlation Cholesky factor and the trained denoiser; Algorithm 2 consumes those artifacts and emits stock-level scenarios, optionally biased toward a regime condition. For temporal generation (Section 3.3) the reverse loop of Algorithm 2 is invoked once per day with the previous day's cross-section as condition, as summarised in Algorithm 3.

Algorithm 1 Training: factor model and factor diffusion

Require: Panel data $\{(R_{i,t}, \mathbf{b}_{i,t})\}$, factor list, hyperparameters $\alpha, T, M, K_z, \ldots$

Stage 1: parametric factor model

$\forall t: \quad \mathbf{F}_t \leftarrow \arg\min_{\mathbf{F}} \sum_i (R_{i,t} - \mathbf{b}_{i,t-1}^\top \mathbf{F})^2$ (3.3)

$\forall i: \quad (a_i, \boldsymbol{\beta}_i) \leftarrow \arg\min \sum_t (R_{i,t+1} - R_t^f - a_i - \boldsymbol{\beta}_i^\top \mathbf{F}_{t+1})^2$ (3.2)

$\forall i: \quad \varepsilon_{i,t} \leftarrow R_{i,t+1} - R_t^f - a_i - \boldsymbol{\beta}_i^\top \mathbf{F}_{t+1}$

$\forall i: \quad (\sigma_i, \nu_i) \leftarrow \mathrm{MLE}\big(\{\varepsilon_{i,t}\}\big)$ (3.4)

Stage 2: factor diffusion

Standardise $\{\mathbf{F}_t\}$; $\mathbf{C} \leftarrow \mathrm{corr}(\mathbf{F})$, $\mathbf{L}\mathbf{L}^\top = \mathbf{C}$

Initialise schedule $\{\gamma_t, \sigma_t, \bar{\gamma}_t, \bar{\sigma}_t\}_{t=1}^T$ from cosine $\bar{\alpha}$ noise shedule

repeat

  Draw minibatch $\{\mathbf{F}^{(b)}\}_{b=1}^B$; $t \sim \mathcal{U}\{1, \ldots, T\}$

  $A^{(m)} \sim S(\alpha/2, 1)$, $m = 1, \ldots, M$ ▷ outer subordinator draws

  $\mathbf{z}^{(m,k)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $k = 1, \ldots, K_z$ ▷ inner Gaussian draws

  $\boldsymbol{\varepsilon}^{(m,k)} = \sqrt{A^{(m)}}\, \mathbf{L}\, \mathbf{z}^{(m,k)}$ (3.10)

  $\mathbf{x}_t^{(m,k)} = \bar{\gamma}_t\, \mathbf{F}^{(b)} + \bar{\sigma}_t\, \boldsymbol{\varepsilon}^{(m,k)}$ (3.7)

  $\ell \leftarrow \mathrm{median}_m\, \mathrm{mean}_k \left\| \hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t^{(m,k)}, t) - \boldsymbol{\varepsilon}^{(m,k)} \right\|^2$ (3.19)

  $\theta \leftarrow \theta - \eta \nabla_\theta \ell$

until converged

return $\{(a_i, \boldsymbol{\beta}_i, \sigma_i, \nu_i)\}_{i=1}^N$, $\hat{\boldsymbol{\varepsilon}}_\theta$, $\mathbf{L}$, $\{\gamma_t, \sigma_t, \bar{\gamma}_t, \bar{\sigma}_t\}$

Finally, Algorithm 3 records the changes needed for temporal generation (Section 3.3): a previous-day condition with dropout at training time, and a Markov-1 rollout that wraps the reverse loop of Algorithm 2 at sampling time. Only the parts that differ from Algorithms 1–2 are shown.

Algorithm 2 Sampling: factor scenarios and stock returns

Require: Trained artifacts from Algorithm 1; optional condition $(\mathbf{c}, \mathcal{L}, s, p)$

$A_t \sim S(\alpha/2, 1),\ t = 1, \dots, T$
$\Sigma_t = \sigma_t^2 A_t + \gamma_t^2 \Sigma_{t-1},\ t = 1, \dots, T$ (3.9)
$\mathbf{x}_T = \sqrt{\Sigma_T}\,\mathbf{L}\mathbf{z},\ \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, T-1, \dots, 1$ **do**
  $\hat{\boldsymbol{\varepsilon}} \leftarrow \hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t)$
  $\Gamma_t = 1 - \gamma_t^2\, \Sigma_{t-1}/\Sigma_t$
  $\boldsymbol{\mu}_t = (\mathbf{x}_t - \bar{\sigma}_t\, \Gamma_t\, \hat{\boldsymbol{\varepsilon}})/\gamma_t$ (3.12)
  if condition supplied and $t > 1$ **then**
    $\hat{\mathbf{x}}_0 = (\mathbf{x}_t - \bar{\sigma}_t\, \hat{\boldsymbol{\varepsilon}})/\bar{\gamma}_t$ (3.15)
    $w_t = \big(\mathrm{SNR}_t/(\mathrm{SNR}_t + 1)\big)^p,\ \mathrm{SNR}_t = \bar{\gamma}_t^2/\bar{\sigma}_t^2$ (3.17)
    $\boldsymbol{\mu}_t \leftarrow \boldsymbol{\mu}_t - s\, w_t\, \Gamma_t\, \Sigma_{t-1}\, \nabla_{\mathbf{x}_t} \mathcal{L}\big(\hat{\mathbf{x}}_0, \mathbf{c}\big)$ (3.16)
  end if
  $\mathbf{x}_{t-1} = \boldsymbol{\mu}_t + \mathbf{1}_{[t>1]} \sqrt{\Gamma_t\, \Sigma_{t-1}}\,\mathbf{L}\mathbf{z},\ \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
end for
$\mathbf{F}_s \leftarrow$ inverse-standardise $\mathbf{x}_0$
for $i = 1, \dots, N$ **do**
  $\varepsilon_{i,s} = \sigma_i \sqrt{(\nu_i - 2)/\nu_i}\, z_{i,s},\ z_{i,s} \sim t_{\nu_i}(0, 1)$ (3.5)
  $R_{i,s} = a_i + \boldsymbol{\beta}_i^\top \mathbf{F}_s + \varepsilon_{i,s}$ (3.2)
end for
return $\{R_{i,s}\}_{i=1}^N$

Algorithm 3 Temporal conditioning: training and Markov-1 sampling (deltas vs. Algorithms 1–2)

Training delta (replaces the minibatch construction of Algorithm 1)
Build consecutive-day pairs $\{(\mathbf{F}_{d-1}, \mathbf{F}_d)\}_d$ from the standardised factor panel
Draw minibatch of pairs $\{(\mathbf{F}_{\text{prev}}^{(b)}, \mathbf{F}^{(b)})\}_{b=1}^B;\ t \sim \mathcal{U}\{1, \dots, T\}$
$\forall b:$ with prob. p_{drop} set $\mathbf{F}_{\text{prev}}^{(b)} \leftarrow \varnothing$ ▷ condition dropout [14]
Form $\mathbf{x}_t^{(b)}$ as in Algorithm 1; condition the denoiser on $\mathbf{F}_{\text{prev}}^{(b)}$ via (3.21)
$\ell \leftarrow \text{median}_m\, \text{mean}_k \left\| \hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t^{(m,k)}, t, \mathbf{F}_{\text{prev}}) - \boldsymbol{\varepsilon}^{(m,k)} \right\|^2$ (3.19)
Sampling delta (Markov-1 rollout, wraps Algorithm 2)
function Reverse($\mathbf{F}_{\text{cond}}$) ▷ lines 1–13 of Algorithm 2, denoiser = $\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t, \mathbf{F}_{\text{cond}})$
  return $\mathbf{x}_0$
end function
$\mathbf{F}_0 \leftarrow$ Reverse($\varnothing$) ▷ self-start from the null condition (3.22)
for $d = 1, \dots, H-1$ **do**
  $\mathbf{F}_d \leftarrow$ Reverse($\mathbf{F}_{d-1}$) (3.23)
end for
return path $\{\mathbf{F}_d\}_{d=0}^{H-1}$ ▷ reconstruct stock returns per day as in Algorithm 2

4

Experiments

This chapter fixes the empirical setup against which the factor-diffusion model of Chapter 3 is assessed. Section 4.1 describes the MSCI World panel and its purely temporal train/out-of-sample split, on which every model is fitted on the training window and judged on unseen 2025 returns. Section 4.2 introduces the two non-learning generators, historical resampling and a parametric Gaussian, which share our factor-decomposition pipeline and differ only in the factor-return sampler, so that any performance gap is attributable to the generative component alone. Section 4.3 reports the denoiser architecture, training and sampling hyperparameters. Section 4.4 then defines the evaluation protocol and the metrics used at the factor and stock levels. Together these sections make the results of Chapter 5 reproducible and the comparison across methods controlled.

4.1. Datasets

The empirical analysis is based on the constituent universe of the MSCI World Index, a free-float market-capitalisation weighted index that tracks the performance of large- and mid-cap equities across 23 developed markets. At the end of our sample the index captures approximately 85% of the free-float adjusted market capitalisation in each country, providing a broad and representative cross-section of the global developed-market equity universe. [1]

For each constituent we obtain two groups of variables from MSCI, total returns in USD at daily frequency. Six MSCI Generic Factors [*growth*, *momentum*, *quality*, *size*, *value* and *volatility*]. Cross-sectionally standardised stock-level exposures produced by MSCI's Global Equity Model and designed to capture the most widely used style premia in the academic and practitioner literature.

Dataset level features are summarised in Table 4.1.

Table 4.1: MSCI World working panel: dataset-level features.

Property	Value
Sample period	2001-01-01 – 2025-12-31 (25 years)
Trading dates	6,523
Unique equities	3,942
Average cross-section	1647 stocks per day (min 968 / max 2104)
Regions	Americas, EMEA, Asia-Pacific
GICS sectors	11 (Financials, Industrials, Cons. Disc. dominant)

The number of constituents grows over the sample, from roughly 1000 names in 2001 to above 2000 names after 2020, reflecting both the expansion of the developed-market universe and successive MSCI inclusion-rule updates. Figure 4.1 (left) visualises this evolution alongside the regional composition of the panel.

[1] MSCI World Index Methodology, `https://www.msci.com/documents/10199/178e6643-6ae6-47b9-82be-e1fc565ededb`.

The six generic factors are constructed at source to be nearly orthogonal style premia, and the right half of Figure 4.1 confirms this in our working sample: each cross-sectional distribution is approximately symmetric around zero with support in $[-3, +3]$, and the time-averaged cross-correlations remain modest. The largest pairwise correlation is between quality and volatility (around 0.33), while every other pair stays inside 0.2 in absolute value, so the six factors carry largely independent information about the cross-section.

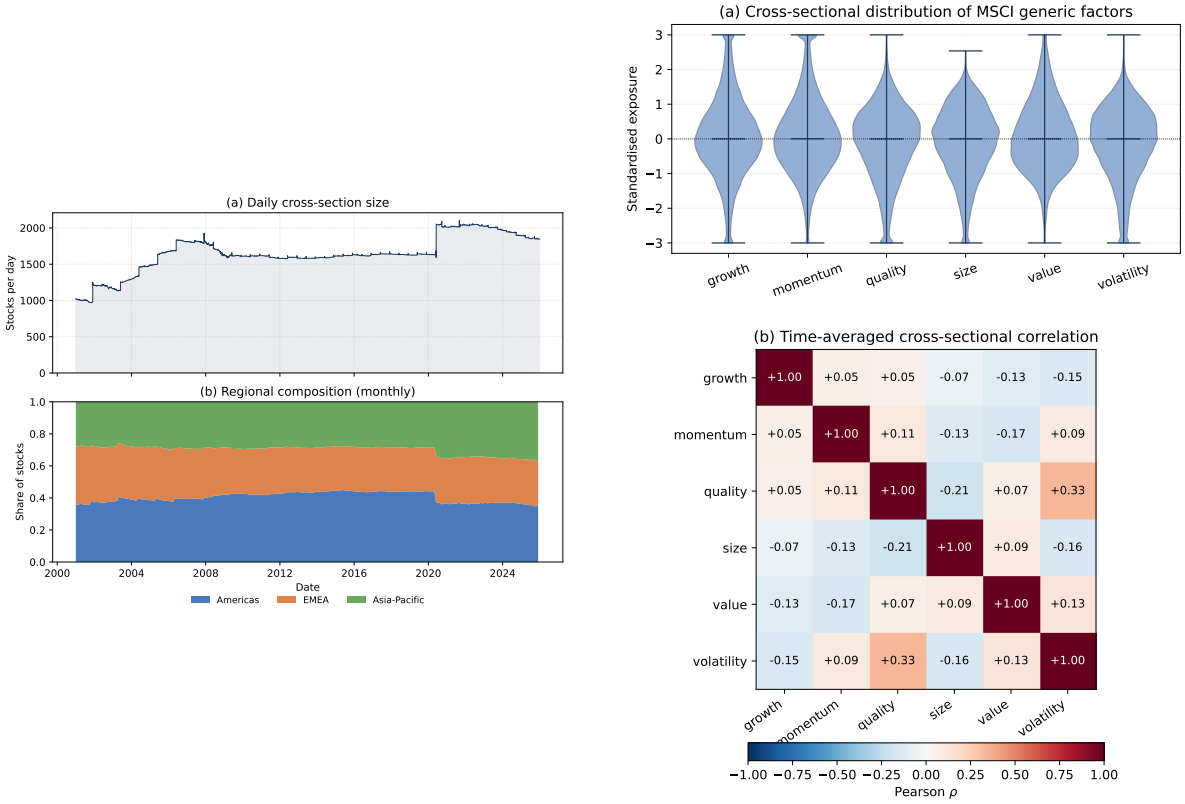


Figure 4.1: MSCI World working panel and factor exposures. **Left:** (a) daily cross-section size and (b) monthly regional composition over 2001-01-01 – 2025-12-31. **Right:** (a) cross-sectional distribution of the six MSCI generic factor exposures and (b) their time-averaged cross-sectional correlation matrix.

Roughly 0.74% of the factor cells are missing in the raw panel, 79417 out of ≈ 10.75 M cells, typically at the start of a firm's listing window or around corporate actions. We impute in two stages. Forward-fill within firm along the time dimension and Cross-sectional median imputation at each date for any cells that remain missing. Because the MSCI generic factors are already winsorised and standardised at source. each factor lies in roughly $[-3, +3]$ with zero mean and unit variance per date, no further outlier treatment is applied.

We retain the last calendar year (2025-01-01 – 2025-12-31, 261 trading days) as an out-of-sample test set and use the remaining 24 years (6262 trading days, 10.26 M rows) for training. The split is purely temporal; 1924 of the 1979 test-set securities are also present in the training set, so the out-of-sample evaluation reflects new return realisations on a largely overlapping cross-section rather than a regime of completely unseen firms.

4.2. Baselines

The contribution of this thesis is the generative model of the factor-return distribution, not the factor decomposition or the idiosyncratic residual model that surround it. To isolate that contribution, every method we compare shares the same pipeline of Section 3.4 and differs only in the block that produces the factor-return draws $\mathbf{F}_s \in \mathbb{R}^K$. The loadings $(a_i, \boldsymbol{\beta}_i)$, the per-stock Student-$t$ residual parameters (σ_i, ν_i), the train/test split, and the inverse-standardisation of the generated factors are held fixed across

all methods; only the sampler that replaces our trained denoiser $\hat{\boldsymbol{\epsilon}}_\theta$ changes. Stock-level scenarios are then reconstructed identically through Equation (3.2), so any difference in downstream performance is attributable to the factor-return generator alone.

We deliberately restrict the comparison to non-learning generators rather than including third-party deep generative models. Architectures such as TimeGAN or return-space GANs operate directly on the joint stock-return or price-path object and are not drop-in replacements for the factor-return block; embedding them in our pipeline would require non-trivial design choices that confound the comparison. The two baselines we retain are instead the canonical reference points for financial scenario generation and precisely the methods the research gap of Chapter 1 targets, namely backward-looking historical resampling and Gaussian Monte Carlo. They bracket our model from two directions: a fully non-parametric generator that reproduces the empirical marginals exactly but cannot extrapolate beyond observed history, and a parametric generator that extrapolates smoothly but assumes thin-tailed, elliptically symmetric co-movements.

Historical resampling (Resample). The first baseline is the empirical bootstrap. Writing $\{\mathbf{F}_t\}_{t=1}^{T}$ for the standardised historical factor-return series, an unconditional cross-sectional draw is an independent sample with replacement from the empirical rows, $\mathbf{F}_s \sim \widehat{\mathbb{P}}_T = \frac{1}{T}\sum_t \delta_{\mathbf{F}_t}$. This reproduces the historical marginal and contemporaneous factor dependence exactly, but by construction it can never generate a factor configuration that was not observed in the training window. For multi-step paths we use the stationary bootstrap of Politis and Romano [26]: a path is grown one step at a time by advancing the current index mod T, and at each step restarting from a uniformly random date with probability $p = 0.125$. The resulting blocks have a geometric length with expectation $1/p = 8$ trading days, which preserves short-horizon autocorrelation and volatility clustering while keeping the resampled path stationary.

Parametric Gaussian (Gaussian). The second baseline replaces the empirical distribution with a multivariate normal fitted to the same series, $\mathbf{F}_s \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$, where $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\Sigma}}$ are the sample mean and sample covariance of the training factor returns. This is the Monte Carlo counterpart to historical simulation: it can produce factor configurations never seen in the data, but only along the directions of an elliptically symmetric, thin-tailed Gaussian, which is exactly the assumption shown to understate equity tail risk [6]. For multi-step paths we let the covariance evolve through the RiskMetrics exponentially weighted moving average [16],

$$\boldsymbol{\Sigma}_t \;=\; \lambda\,\boldsymbol{\Sigma}_{t-1} \;+\; (1-\lambda)\,\mathbf{d}_t\mathbf{d}_t^{\top}, \qquad \mathbf{d}_t = \mathbf{F}_t - \hat{\boldsymbol{\mu}}, \tag{4.1}$$

with decay $\lambda = 0.94$ and the recursion warm-started at the sample covariance $\hat{\boldsymbol{\Sigma}}$. Each path step is drawn from $\mathcal{N}(\hat{\boldsymbol{\mu}}, \boldsymbol{\Sigma}_t)$, giving the baseline a GARCH-like time-varying second moment while keeping every conditional distribution Gaussian.

Conditional sampling. Both baselines admit the same regime-conditioning interface as our model, so the conditional evaluation of Section 5.2 is well defined for all three. Lacking a reverse diffusion process to steer, the baselines realise the conditioning loss $\mathcal{L}$ by rejection sampling: a candidate $\mathbf{F}$ is generated from the unconditional sampler, standardised with the shared scaler, and accepted with probability $\exp\big(-s\,\mathcal{L}(\mathbf{F})\big)$, using the same guidance scale s as the diffusion model. This targets the same tilted distribution $\propto \widehat{\mathbb{P}}(\mathbf{F})\,\exp(-s\,\mathcal{L})$ that the guided diffusion of Algorithm 2 approximates, which keeps the conditional comparison on a common footing.

4.3. Implementation Details

We adopt a medium-complexity Transformer denoiser: each of the seven MSCI factors enters as a separate token, six Transformer blocks with eight attention heads each, and an AdaLN layer that injects the diffusion timestep.

Training operates on standardised factors: a scaler fit on the training set normalises each factor to zero mean and unit variance and is reused at sampling time. Optimisation uses AdamW with weight

decay and gradient clipping at norm 1.0; both have little effect in well-behaved batches but absorb the occasional large gradients produced by heavy-tailed noise.

On the schedule side, the stability index $\alpha = 1.9$ preserves the extreme moves characteristic of factor returns, while clipping the subordinator A_t to $[0, 2000]$ keeps rare astronomical draws from corrupting the schedule arithmetic. The noise schedule itself is the cosine $\bar{\alpha}_t$ of [24], which spreads useful learning signal across the full diffusion trajectory rather than concentrating it at one end.

The Lévy training loss is estimated by a median-of-means scheme with five outer and five inner Monte Carlo draws per minibatch, again for stability since a single subordinator draw can be extreme. At sampling time we generate 4096 cross-sections per run; conditional sampling adds a guidance gradient whose strength decays with the per-step signal-to-noise ratio, so guidance acts strongly when there is still signal to steer and fades out near the noise floor. Table 4.2 summarises the hyperparameters of the factor diffusion model.

Table 4.2: Factor diffusion hyperparameters.

Group	Parameter	Value
Denoiser	Token embedding dim d	128
	Number of Transformer blocks	6
	Attention heads	8
	Timestep embedding dim	128
Training	Optimiser	AdamW
	Learning rate	1×10^{-4}
	Weight decay	1×10^{-4}
	Epochs	200
	Batch size	64
Scheduling	Diffusion steps T	100
	Schedule shape	Cosine $\bar{\alpha}_t$
	Stability index α	1.9
	Subordinator clip	$A_t \in [0, 2000]$
Loss	Outer MC draws M	5
	Inner MC draws K_z	5
Sampling	Samples per evaluation	4096
	Conditional guidance scale s	5.0
	SNR weight p	1.0

To obtain the factor return time series consumed by the diffusion model, we apply two standard estimators in parallel: a cross-sectional regression and a univariate portfolio sort. Figure 4.2 reports the resulting cumulative factor returns under both methods.

The two estimators agree on the sign of every premium and identify growth, momentum and value as the most rewarded styles. Portfolio-sort spreads are 2 to 3 times larger in magnitude, but they are also substantially more volatile, and only growth remains significant at the 5% level. The regression estimator, by hedging exposures across all six factors simultaneously, produces lower-variance returns and recovers three significant premia.

All following evaluation uses the regression-based factor returns. A final relabelling is applied: the cross-sectional intercept has the economic interpretation of the equally-weighted return left after removing the six style premia. We rename it the market factor, which is how the original six MSCI generic factors become the seven factors used as model inputs throughout this thesis.

4.4. Evaluation Metrics

We assess generation quality along two cross-sectional levels of aggregation, the low-dimensional factor returns and the high-dimensional stock cross-section, and, for the temporal generation task, along the time dimension of individual stock paths. This section defines the metric used at each level together with the reason it is included; metrics are chosen on two grounds, that they are either widely adopted in

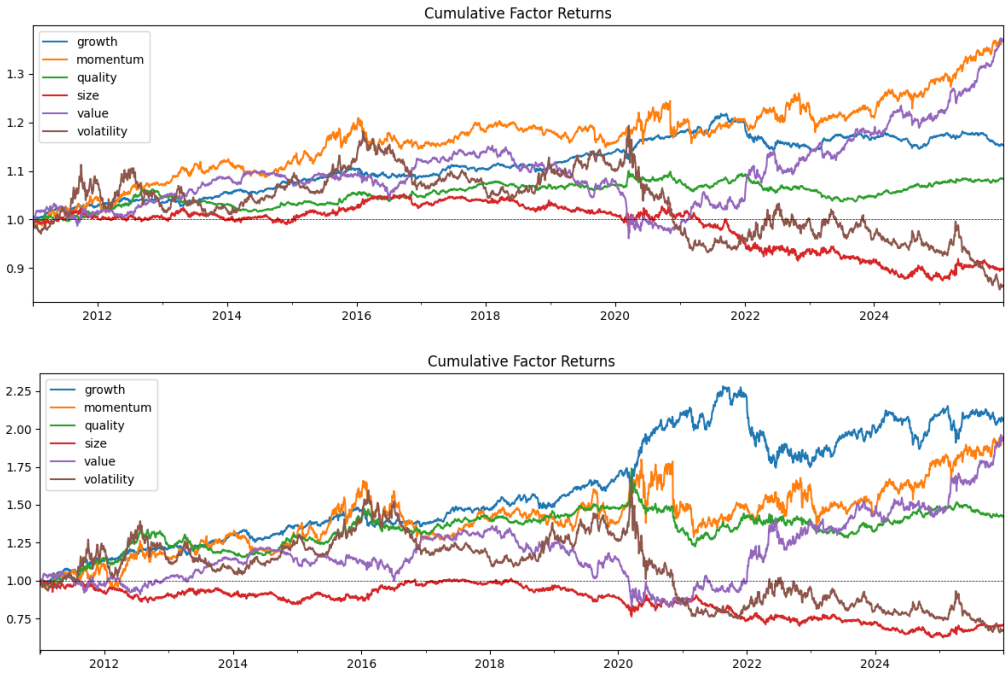


Figure 4.2: Cumulative factor returns over 2001–2025 under the cross-sectional regression estimator (top) and the quintile portfolio-sort estimator (bottom).

the distributional-comparison and financial-econometrics literature, or directly motivated by a stylized fact of equity returns [6] such as heavy tails or negative skew. All formulas are collected in Table 4.3.

At the factor level we begin with the Mahalanobis distance: relative to a reference mean $\hat{\boldsymbol{\mu}}$ and covariance $\hat{\boldsymbol{\Sigma}}$, a draw's squared Mahalanobis distance would follow a χ^2_K law if the generated factors were multivariate Gaussian, so a QQ plot of the ordered distances against the χ^2_K quantiles tests joint normality. Because equity returns are heavy-tailed [6], the distances should curve above the diagonal in the right tail, and a generator that keeps Gaussian-like joint tails would instead sit on the diagonal, which is where the elliptical Gaussian baseline is expected to break down.

Each factor's marginal distribution is then summarised by its first four moments the mean for location, the standard deviation for dispersion, the skewness for the downside asymmetry typical of equity factors, and the excess kurtosis for the fat tails that are a defining stylized fact [6] and, because four moments do not pin down the full shape, by the Wasserstein-1 distance, the integrated absolute difference of two CDFs, which responds to the entire distribution rather than to any finite set of moments. Cross-factor dependence is captured by the correlation matrix and summarised by the relative Frobenius error between two correlation matrices: co-movement drives diversification and joint tail risk, so a generator can match every marginal yet remain useless if it distorts the dependence structure, and the relative Frobenius error collapses the whole-matrix discrepancy into a single scale-free number.

At the stock level the cross-section is summarised rather than inspected name by name. The same four time-series moments are computed per stock, with the standard-deviation panel the key check that individual stock volatilities are reproduced rather than averaged away by aggregation. A per-stock two-sample Kolmogorov–Smirnov test yields a p-value p_n for each name; the KS test is a widely adopted nonparametric test of whether two samples share a distribution, and aggregating across stocks turns it into a population statement, since under the null of identical distributions the $\{p_n\}$ are uniform on $[0, 1]$ — a histogram piling up near zero exposes systematic mismatch, and the fraction above 0.05 counts how many stocks pass. Finally, because each return decomposes into a systematic factor component and an idiosyncratic residual, the cross-sectional dispersion is split accordingly and, per component, summarised by a dispersion ratio and the Wasserstein-1 distance between two dispersion series. Separating systematic from idiosyncratic dispersion is standard practice in cross-sectional asset pricing [10] and makes the diagnosis actionable: an inflated cross-section can be localised to factor

returns or loadings that are too volatile, or to an overstated residual variance, which a single aggregate error could never separate.

The temporal generation task is judged along the time dimension of each simulated path, where the benchmarks are the stylized facts of return dynamics rather than the shape of any single marginal. We adopt the evaluation suite of Kubiak et al. [19], computing each metric per scenario per asset on the series and averaging over the scenarios and assets. The first three measures test the autocorrelation stylized facts catalogued by Cont [6]. The return autocorrelation gauges linear predictability: efficient-market returns carry no exploitable serial correlation, so a faithful generator keeps it near zero. The volatility-clustering measure is the autocorrelation of absolute returns, which is positive and slowly decaying in real markets because large moves cluster in time. The coarse fine measure is the lead–lag cross-correlation between coarse and fine grained volatility; its asymmetry encodes the volatility-cascade effect, whereby long-horizon volatility predicts short-horizon volatility more strongly than the reverse, and is expected to be positive. The variance ratio of Lo and MacKinlay [20], contrasts the realised day cumulative variance with the random-walk prediction: a value of one signals a memoryless walk, above one momentum or trend, and below one mean reversion. Finally the maximum drawdown records the largest peak-to-trough loss along the cumulative-return path, a path-dependent tail measure that risk managers care about directly and that no marginal moment can recover.

Table 4.3: Evaluation metrics used in this thesis. For a sample $\{x_i\}_{i=1}^n$, $\bar{x}$ and σ are its mean and standard deviation; P, Q denote the two samples or estimates being compared, F_P a CDF and $\hat{F}_P$ an empirical CDF.

Metric	**Formula**
Marginal moments	$\bar{x} = \frac{1}{n}\sum_i x_i, \quad \sigma = \sqrt{\frac{1}{n}\sum_i (x_i - \bar{x})^2}, \quad \gamma_1 = \mathbb{E}\left[\left(\frac{x-\bar{x}}{\sigma}\right)^3\right], \quad \gamma_2 = \mathbb{E}\left[\left(\frac{x-\bar{x}}{\sigma}\right)^4\right] - 3$
Wasserstein-1	$W_1(P,Q) = \int_{-\infty}^{\infty} \left\lvert F_P(x) - F_Q(x)\right\rvert dx$
Mahalanobis Q–Q	$d^2(\mathbf{F}) = (\mathbf{F} - \hat{\boldsymbol{\mu}})^\top \hat{\boldsymbol{\Sigma}}^{-1} (\mathbf{F} - \hat{\boldsymbol{\mu}})$, ordered against χ^2_K quantiles
Correlation error	$\rho_{ij} = \dfrac{\mathrm{Cov}\left(F^{(i)}, F^{(j)}\right)}{\sigma_i \sigma_j}, \qquad \mathrm{Err}_F = \dfrac{\left\lVert \boldsymbol{\rho}_P - \boldsymbol{\rho}_Q \right\rVert_F}{\left\lVert \boldsymbol{\rho}_Q \right\rVert_F}$
Kolmogorov–Smirnov	$D = \sup_x \left\lvert \hat{F}_P(x) - \hat{F}_Q(x)\right\rvert$; report the distribution of $\{p_n\}$ and $\frac{1}{N}\sum_n \mathbb{1}[p_n > 0.05]$
Dispersion comparison	$\mathrm{CSD}_t^{(c)} = \mathrm{std}_n\, r_{n,t}^{(c)}$, $r_{n,t}^{(\mathrm{sys})} = \boldsymbol{\beta}_n^\top \mathbf{F}_t$, $r_{n,t}^{(\mathrm{idio})} = \varepsilon_{n,t}$; report $R^{(c)} = \overline{\mathrm{CSD}}_P^{(c)} / \overline{\mathrm{CSD}}_Q^{(c)}$ and W_1 of the $\{\mathrm{CSD}_t^{(c)}\}$ series, $c \in \{\mathrm{total}, \mathrm{sys}, \mathrm{idio}\}$
Autocorrelation (ACF)	$\mathrm{ACF}(k) = \mathrm{Corr}\left(r_t,\, r_{t-k}\right)$; report the lag-averaged $\frac{1}{K}\sum_{k=1}^K \mathrm{ACF}(k)$, $K = 10$
Volatility clustering	$\mathrm{VolClust}(k) = \mathrm{Corr}\left(\lvert r_t\rvert,\, \lvert r_{t-k}\rvert\right)$; report the lag-averaged value, $K = 10$
Coarse–fine	$c_t = \left\lvert \sum_{s=t-\tau+1}^{t} r_s \right\rvert, \quad f_t = \sum_{s=t-\tau+1}^{t} \lvert r_s\rvert, \quad \overline{\mathrm{CF}} = \dfrac{1}{K}\sum_{k=1}^{K} \mathrm{Corr}\left(c_{t+k},\, f_t\right), \quad \tau = 5$
Variance ratio	$\mathrm{VR}(h) = \dfrac{\mathrm{Var}\left(\sum_{i=1}^h r_{t-i+1}\right)}{h\ \mathrm{Var}(r_t)}$
Maximum drawdown	$V_t = \prod_{s \le t}(1 + r_s), \qquad \mathrm{MDD} = \max_t \dfrac{\max_{s \le t} V_s - V_t}{\max_{s \le t} V_s}$

5

Results and Evaluation

5.1. Unconditional Cross-Sectional Generation

We begin by asking whether the model, sampled without any conditioning, reproduces the unconditional statistical structure of the factor cross-section.The marginal and joint distributions it produces have to match those of the real returns. Throughout we compare the generated sample against two baselines, an empirical Resample of the historical cross-section and an elliptical Gaussian fit, all evaluated on the same out-of-sample (OOS) period.

We organise the comparison from the joint distribution inward to its components. Section 5.1.1 checks the multivariate tail through the Mahalanobis radius. Sections 5.1.2 and 5.1.3 turn to the per-factor marginals, summarising them through their first four moments and through full densities, Wasserstein distances and quantile plots. Finally, Section 5.1.4 examines the cross-sectional correlation structure that ties the seven factors together.

5.1.1. Joint Tail Behaviour

We first check that the target itself carries the stylized fact this work is built around: the heavy tails of equity factor returns. We use the squared Mahalanobis distance, which collapses the seven-dimensional cross-section into a single radial coordinate. Under a multivariate Gaussian d^2 follows a χ^2_7 law, so a Q–Q plot of the sorted distances against χ^2_7 quantiles lies on a straight line; any upward curvature in the upper tail signals joint tails heavier than Gaussian.

The left panel of Figure 5.1 confirms the stylized fact: the OOS distances track the χ^2_7 line near the centre but peel away above it from roughly the 10th quantile onward, reaching $d^2 \approx 60$ where the Gaussian reference predicts under 20, the signature of a heavy joint tail. The right panel shows that the generated sample inherits exactly this curvature, climbing alongside the Resample baseline into the extreme quantiles, while the Gaussian baseline flattens onto the reference line and tops out near $d^2 \approx 25$. The heavy-tailed forward process therefore reproduces the multivariate tail of the cross-section that an elliptical-Gaussian model cannot.

5.1.2. Statistical Moments

We first summarise each factor's marginal distribution by its first four moments, defined in Table 4.3. Figure 5.2 plots, for each of the seven factors, the moment of the generated and baseline samples against the corresponding out-of-sample (OOS) value, so that a point on the dashed diagonal denotes an exact match and the per-factor spread of each method is visible directly. Table 5.1 then aggregates this comparison into a single mean absolute error (MAE) across factors, with the across-factor standard deviation in parentheses.

The three methods coincide on the mean and the standard deviation and separate only on the higher moments, which is where the heavy tails and downside asymmetry of equity returns [6] live and where a generator is actually tested. Location and scale are therefore a sanity check that every method clears

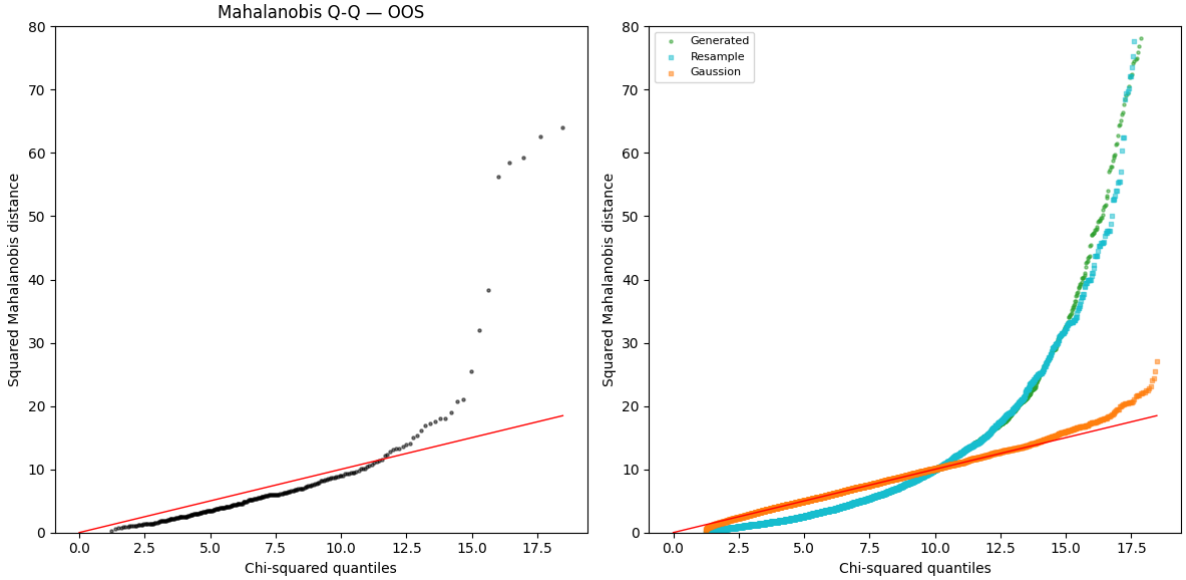


Figure 5.1: Mahalanobis Q–Q plots of the squared distance d^2 against χ^2_7 quantiles; the red line is the Gaussian reference. *Left:* the OOS sample bends sharply above the reference in the upper tail. *Right:* the generated (green) and Resample (cyan) samples reproduce this upward curvature, whereas the Gaussian baseline (orange) stays on the reference line and cannot reach the extreme distances.

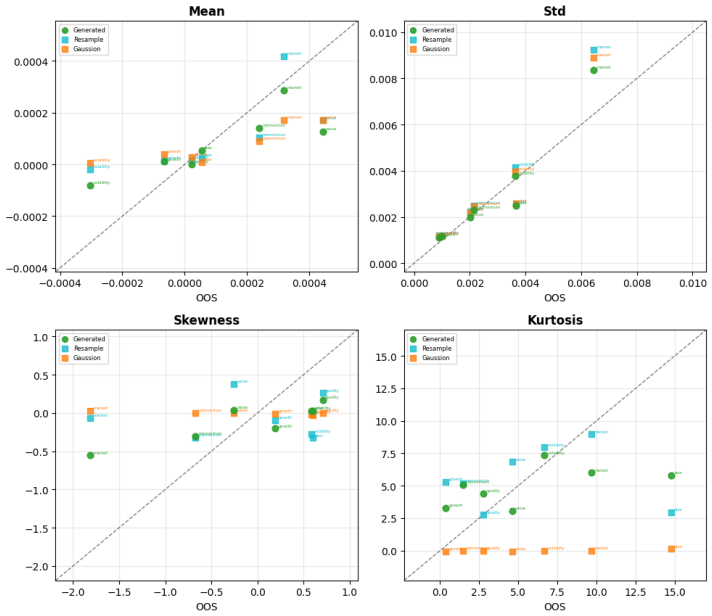


Figure 5.2: Marginal-moment comparison at the factor level. Each panel plots one moment (mean, standard deviation, skewness, excess kurtosis) of the generated (green) and baseline (Resample, Gaussian) samples on the vertical axis against the OOS value on the horizontal axis, one point per factor; points on the dashed diagonal match the OOS moment exactly.

Table 5.1: Per-moment MAE (↓, lower is better) across the seven factors between each method and the OOS sample, as mean (standard deviation) over factors. Row-best in bold.

	Gen. ↓	Resamp. ↓	Gauss. ↓
Mean	**0.0001** (0.0001)	**0.0001** (0.0001)	**0.0001** (0.0001)
Std	**0.0005** (0.0007)	0.0008 (0.0009)	0.0007 (0.0008)
Skew	**0.5687** (0.2966)	0.7504 (0.4658)	0.7006 (0.4994)
Kurt	**3.2859** (2.5528)	3.5267 (3.7421)	5.7692 (4.6579)

rather than a point of discrimination, and the comparison that matters is the third and fourth moments.

On both, Figure 5.2 shows the Gaussian baseline collapsing onto the zero line irrespective of the OOS value: an elliptical Gaussian is symmetric and thin-tailed by construction, so it cannot represent a non-zero skewness or a positive excess kurtosis at all. The generated sample instead rises with the OOS value across factors, recovering both the sign of the asymmetry and the fattening of the tails; this is the direct payoff of the heavy-tailed forward process, which lets the model place mass in regions an elliptical-Gaussian baseline can never reach.

The across-factor standard deviations for skewness and excess kurtosis, of the same order as the errors themselves, so the higher-moment fit is far less uniform across the seven factors than the location and scale fit. Relative to the mean and the standard deviation which every method recovers almost exactly the third and fourth moments are therefore only approximately matched which reflects how much harder the tail of the cross-section finance scenario is to module than its centre.

5.1.3. Marginal Distributions

Figure 5.3 overlays, for every factor, a Gaussian kernel-density estimate of the OOS sample against the generated and the two baseline samples, so that the location, the peak and the tails of each marginal can be compared at once. Table 5.2 then reduces each overlay to a single number, the first Wasserstein (W_1) distance to the OOS marginal (Table 4.3), and Figure 5.4 inspects the quantiles directly through a per-factor Q–Q plot against the OOS sample.

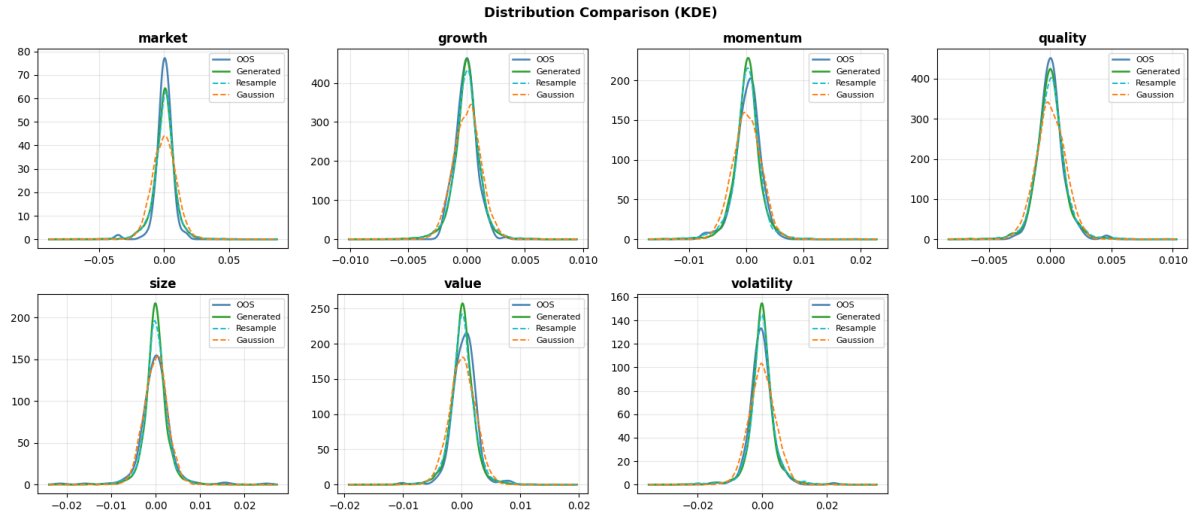


Figure 5.3: Per-factor marginal densities. Each panel shows a Gaussian kernel-density estimate of the OOS (blue), generated (green), Resample (dashed) and Gaussian (dashed) samples for one factor. The generated density tracks the OOS peak and shoulders across the seven factors, whereas the Gaussian baseline is visibly too peaked at the centre and too light in the shoulders.

Across the seven factors the generated density sits almost on top of the OOS density: it matches the mode, the width and the asymmetry of the shoulders, which is the marginal counterpart of the higher-moment fit in Section 5.1.2. The Gaussian baseline, by contrast, is systematically over-concentrated, it places too much mass at the centre and too little in the shoulders, exactly the failure expected from a thin-tailed elliptical fit. The clearest case is the *market* factor, whose OOS marginal is the most heavy-shouldered of the seven: there the Gaussian peak overshoots the OOS density while the generated and Resample curves stay close to it.

The W_1 distances in Table 5.2 make this quantitative. The generated sample attains the lowest distance on five of the seven factors and ties on the rest, giving the best across-factor average. The gap is widest precisely where the marginals are hardest: on market the generated distance (0.0017) is roughly half the Gaussian one (0.0032), confirming that the heavy-tailed forward process pays off most on the heavy-shouldered factor, while on the near-Gaussian factors (growth and quality) all three methods are essentially indistinguishable.

Finally, the Q-Q plots in Figure 5.4 examine the marginals quantile by quantile. The generated quantiles

Table 5.2: First Wasserstein distance (W_1, ↓ lower is better) between each method's marginal and the OOS marginal, per factor and averaged across factors. Row-best in bold.

Factor	Gen. ↓	Resamp. ↓	Gauss. ↓
market	**0.0017**	0.0019	0.0032
growth	**0.0002**	**0.0002**	**0.0002**
momentum	**0.0003**	**0.0003**	0.0005
quality	**0.0001**	0.0002	0.0002
size	0.0005	**0.0004**	0.0005
value	**0.0003**	0.0004	0.0005
volatility	**0.0005**	**0.0005**	0.0008
Average	**0.0005**	0.0006	0.0008

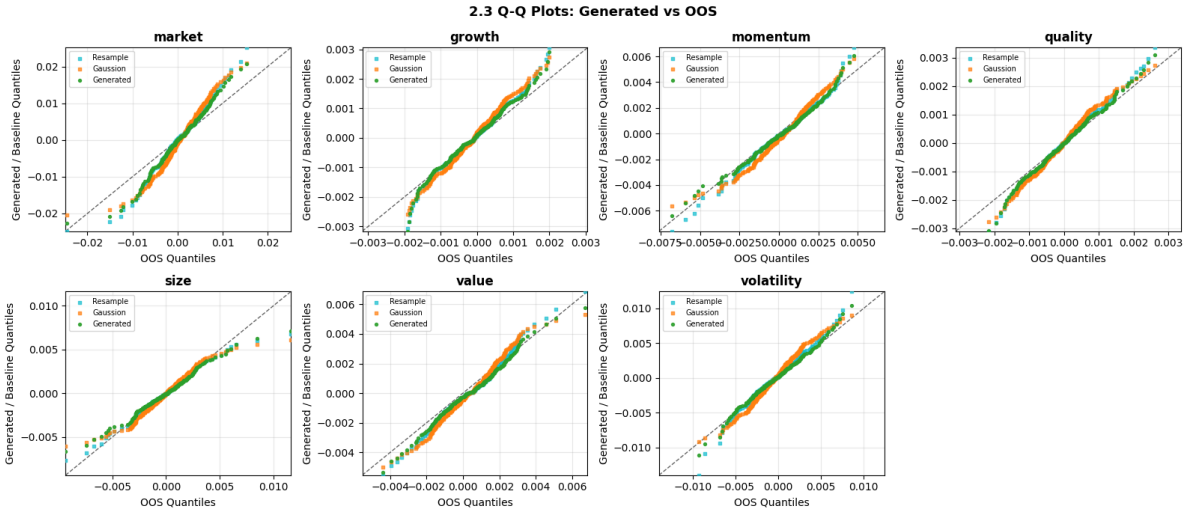


Figure 5.4: Per-factor quantile–quantile plots against the OOS sample. Each panel plots 200 evenly spaced quantiles of the generated (green), Resample and Gaussian samples against the corresponding OOS quantiles; points on the dashed diagonal indicate an exact quantile match. The generated quantiles lie on the diagonal across the full support, including the tails, where the baselines start to deviate.

fall on the diagonal across the whole support and, stay on it into the extreme quantiles, where a marginal-distance summary is least informative and where the Gaussian baseline bends away from the line.

5.1.4. Factor Correlations

We now turn to the cross-sectional correlation structure to show the linear dependence between factors. Figure 5.5 shows the correlation heatmaps for the OOS and training samples next to those of the three methods, and Figure 5.6 split each matrix to its 21 off-diagonal entries and plots them against the corresponding OOS correlations.

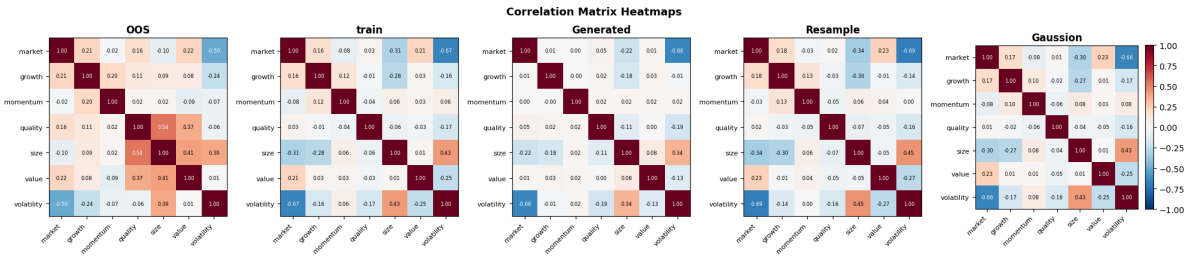


Figure 5.5: Factor correlation matrices for the OOS and training samples and for the generated, Resample and Gaussian samples.

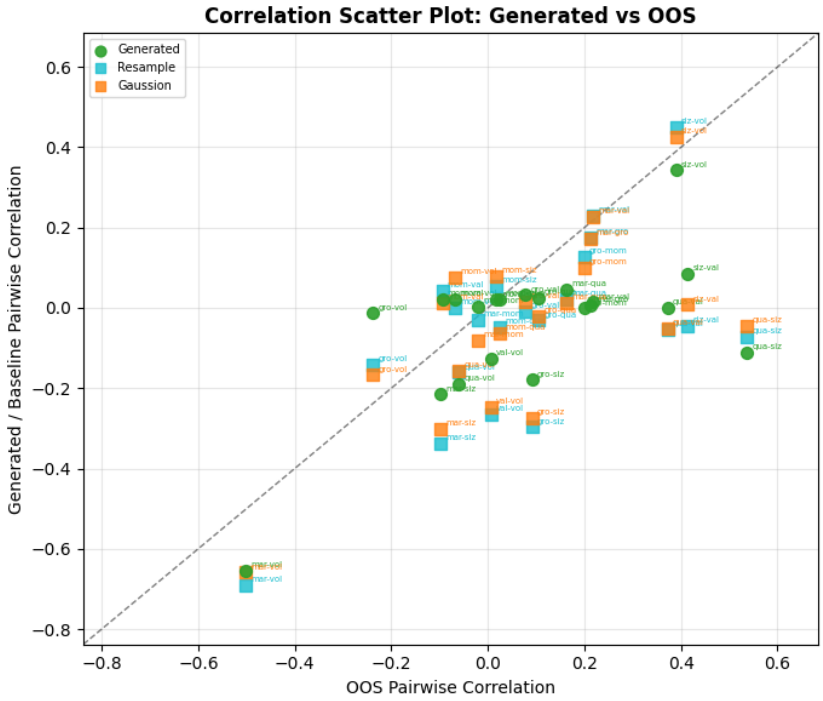


Figure 5.6: Off-diagonal correlations of each method (vertical axis) against the OOS correlations (horizontal axis), one point per factor pair; points on the dashed diagonal match the OOS correlation exactly.

In relative MAE against the OOS covariance, our method is the best of the three, at 0.1674 (generated) versus 0.1738 (Resample) and 0.1687 (Gaussian), though the margin is small. More striking in Figures 5.5–5.6 is the visible gap to the OOS structure: the scatter points sit mostly below the diagonal, meaning the generated correlations are weaker than the OOS ones. The reason is a regime shift between the two periods. Several factor correlations, in particular within the quality–size–value block, are markedly stronger in the OOS sample than in the training sample (e.g. quality–size rises from 0.54 in OOS while it is near zero in training). The model is trained on the training period and can only reproduce its weaker dependence, and every method sits below the diagonal on the same pairs.

5.2. Conditional Cross-Sectional Generation

Having established that the unconditional model reproduces the cross-section, we now ask whether it can be steered into a prescribed stress regime. We investigate not only whether the targeted factor moves, but whether the joint seven-factor response, the induced shifts in the other factors and in the dependence structure, matches what the regime looks like in reality.

We read every conditional quantity against four references. The unconditional sample is the un-steered baseline. Soft rejection (Section 3.2.2) is rejection sampling on conditional loss. The Gaussian baseline applies the same condition to an MVN gaussion sample, and the historical stress days, the real out-of-sample days that satisfy the constraint, are the ground-truth regime that all generators are ultimately trying to reproduce. Throughout we generate N=1000 conditional samples per method.

5.2.1. Volatility Conditioning

We take as the worked example a low-volatility-factor stress, descripted as volatility less than 1% percentage of unconditional volatility, and trace its effect from the targeted marginal outward to the full joint distribution. The historical analogue of this regime is the out-of-sample days on which the volatility factor fell below the same threshold.

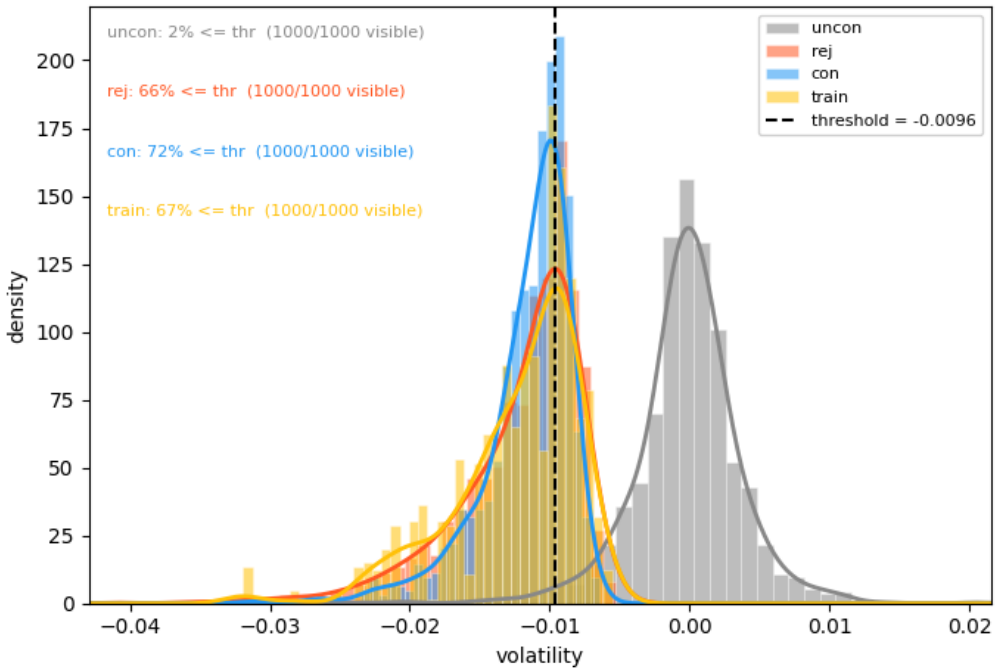


Figure 5.7: Marginal density of the targeted volatility factor under the condition volatility < -0.0096 (dashed line). The unconditional sample (grey) places only 2% of its mass in the target tail; guidance (blue) lifts this to 72%, matching soft rejection (66%) and the resampled historical analogue (67%).

Figure 5.7 shows a sanity check of the guidance method. Only 2% of the unconditional mass lies below the threshold, by construction, since this is the 1% tail; under guidance 72% of the conditional sample clears it. The decisive comparison is against soft rejection, which concentrates 66% of its mass in the same region, and against the resampled historical analogue at 67%. The similar shape between the guidance and soft-rejection KDEs confirms that the guidance approximation is accurate.

As another representation. Figure 5.8 shows marginal density of all seven factors in a violin plot. Frome the rejection baseline, We see a structured cross-factor spillover rather than an isolated shift. for example, pushing the volatility factor down to a median near -0.011 pulls the market factor up to roughly +0.017 and stretchs the size and value factors. and since the guidance violins sit almost exactly on top of the soft-rejection violins, Our results inherit the dependence-driven response.

We plot each method's pairwise factor correlations against those of the historical stress days to evaluate coditioning factor comovement 5.9 . Summarising the whole matrix by the relative Frobenius

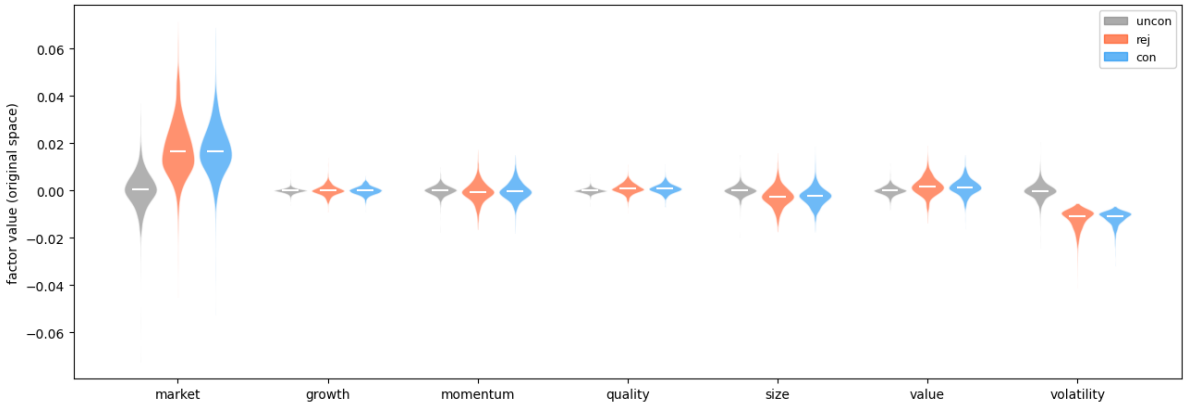


Figure 5.8: Joint response of all seven factors to the volatility condition. Violins show the unconditional (grey), soft-rejection (orange) and guidance (blue) samples; the white bar marks the median. Conditioning the volatility factor alone shifts the market factor upward and tilts size and value, and the guidance violins coincide almost exactly with the soft-rejection violins on every factor.

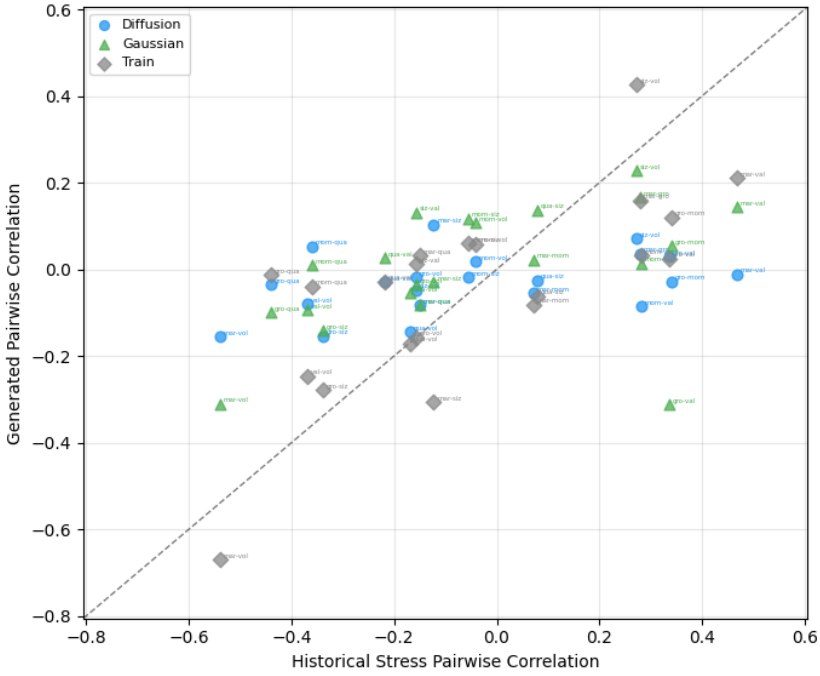


Figure 5.9: Pairwise factor correlations of each method (vertical axis) against the historical-stress correlations (horizontal axis), one point per factor pair; points on the dashed diagonal match the historical-stress dependence exactly.

error of Table 4.3 against the historical-stress covariance, guidance attains 0.5066, below both the Gaussian baseline (0.7943) and the unconditional training cross-section (0.6706). The guided conditional dependence is therefore closer to the historical stress regime than the raw training correlations. Our conditioning does not merely preserve the unconditional dependence, it sharpens it toward the regime, which an mutivariate Gaussian, whose correlation structure is fixed by its single covariance, cannot do.

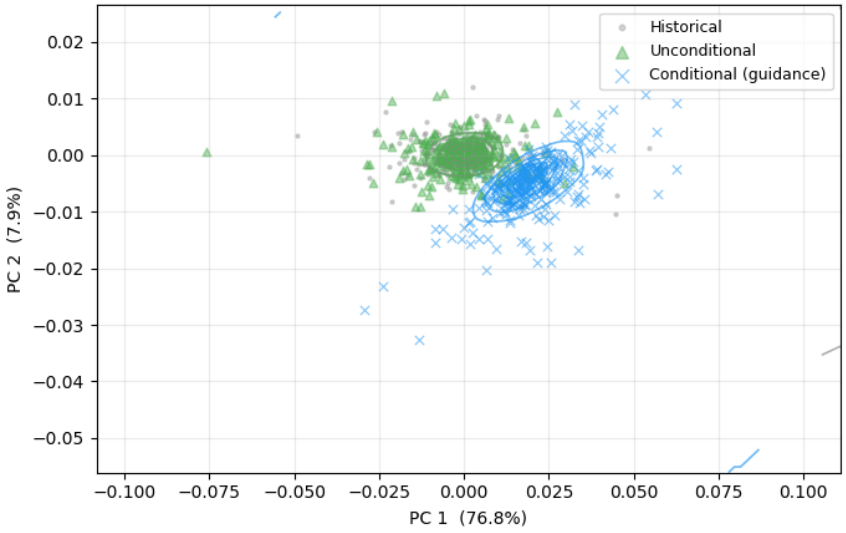


Figure 5.10: Projection of the unconditional and conditional factor scenarios onto the first two principal components of the historical factor cross-section (76.8% and 7.9% of variance). The guidance cloud (blue) detaches coherently from the unconditional cloud (green) while staying inside the historical envelope (grey).

Finally, Figure 5.10 views the effect globally, projecting the samples onto the first two principal components of the historical cross-section. The conditional cloud separates cleanly from the unconditional one along a single coherent direction rather than smearing in every direction, and it remains within the support of the historical scenarios. The condition therefore relocates the joint distribution as a body, consistent with the marginal, spillover and dependence evidence above, instead of carving an unconditional sample down to its target tail.

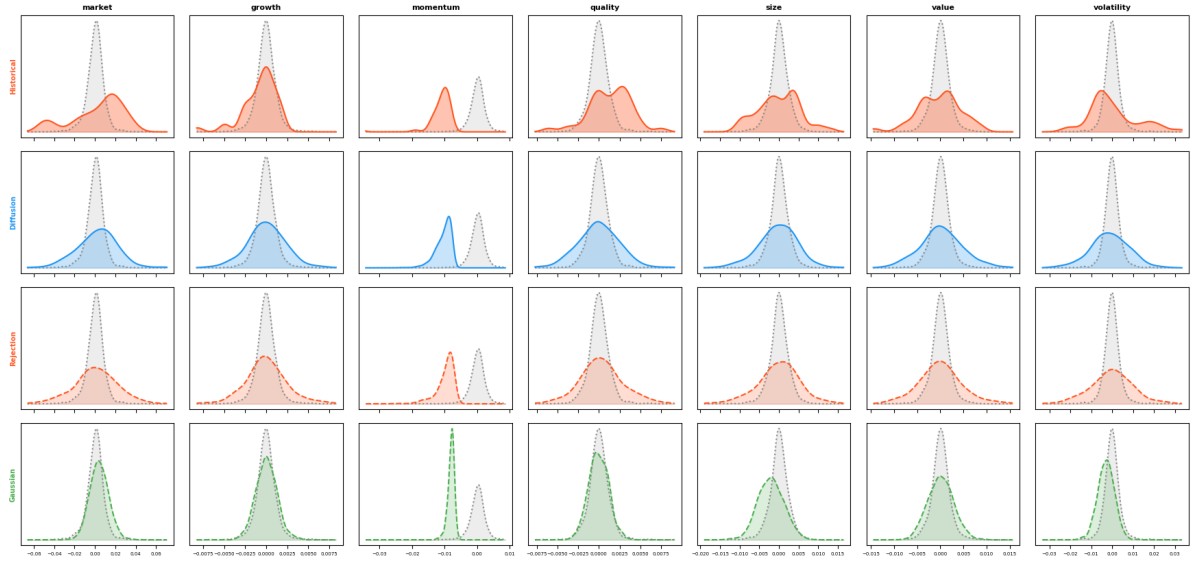
5.2.2. Momentum Conditioning

The same machinery should work on any factor and in either direction. We test both with momentum: a downside momentum-crash condition on its 1% quantile, and an upside momentum-rally condition on its 99% quantile. Figures 5.11a and 5.11b show the per-factor conditional densities for the two regimes; the remaining diagnostics, which repeat the volatility analysis, are collected in Appendix A.3.

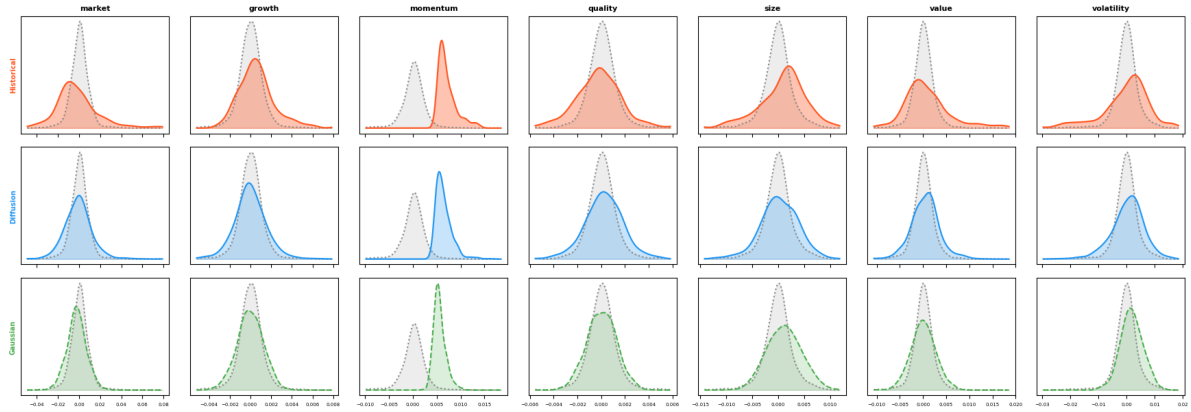
Under the crash condition guidance concentrates 81% of its mass below the threshold, bracketing soft rejection 64% and the historical analogue 59%, and the per-factor densities of Figure 5.11a show the guidance row tracking the historical-stress row, with the momentum marginal sharply displaced into its lower tail while the other factors move only as their dependence with momentum dictates. The rally condition is the mirror image: 69% of the guided mass now lies above the threshold, against 55% historically, and Figure 5.11b shows the same displacement reflected into the upper tail. Guidance is thus direction-agnostic, it steers a factor up as readily as down, and the mechanism transfers unchanged from the volatility factor to momentum.

The conditions above are imposed at the factor level, but a scenario generator is ultimately used on individual stocks. Reconstructing stock returns from the conditional factors through the factor model lets us check that a factor stress propagates correctly to the cross-section. The mechanism is a loading effect: a stock's conditional expected return should move in proportion to its loading on the stressed factor, so a regression of conditional expected stock return on the momentum loading β should have a slope whose sign and magnitude encode how hard the regime hits momentum-exposed names.

Figure 5.12 confirms the loading effect and its direction. Under the momentum crash the relationship is downward sloping, high-momentum stocks lose the most, with a guidance slope of -0.0101 that sits



(a) Momentum-crash regime



(b) Momentum-rally regime

Figure 5.11: Per-factor conditional densities for the momentum regimes: the historical stress days, guidance Diffusion, soft rejection and the Gaussian baseline (rows), against the unconditional baseline (dotted), one column per factor.

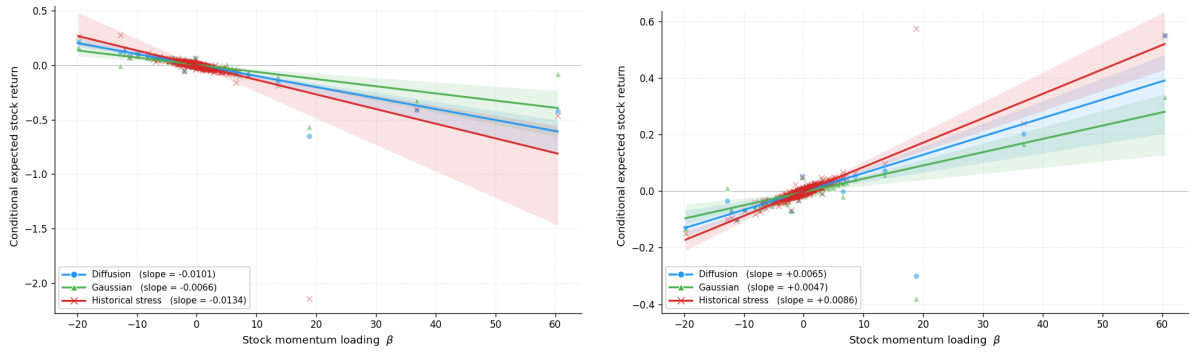
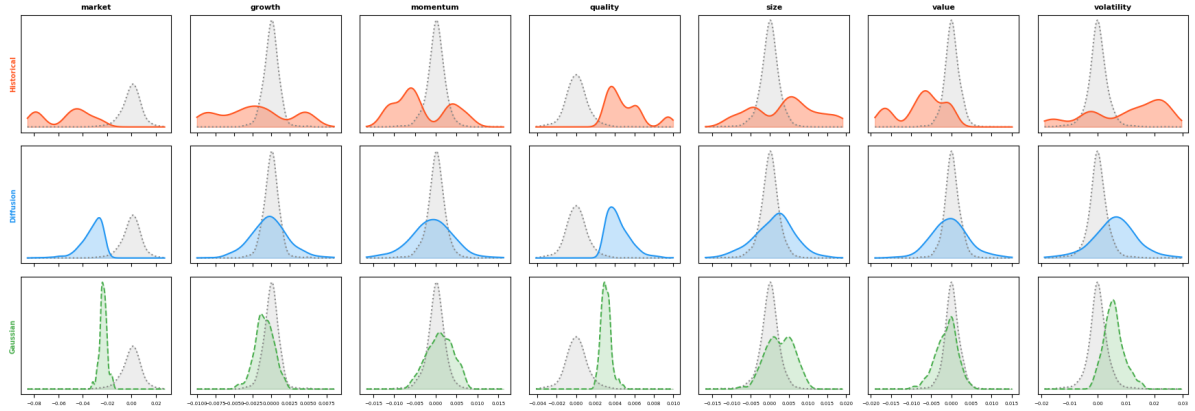


Figure 5.12: Conditional expected stock return against the stock's momentum loading β, under the momentum-crash (left) and momentum-rally (right) regimes, for guidance Diffusion, the Gaussian baseline and the historical stress days. The fitted slope measures how strongly the factor stress propagates to momentum-exposed stocks.

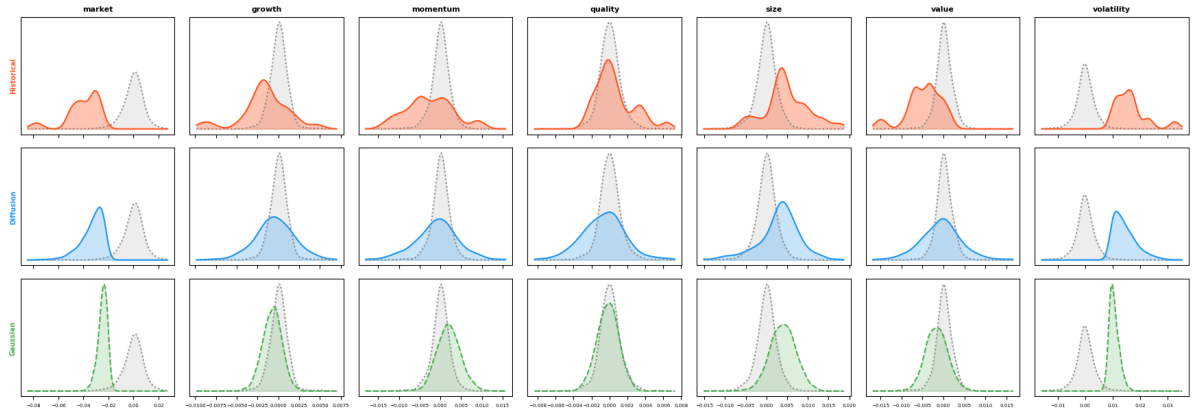
between the Gaussian baseline (-0.0066) and the historical stress days (-0.0134); under the rally the slope flips positive, +0.0065 for guidance against +0.0047 (Gaussian) and +0.0086 (historical). In both directions guidance recovers the sign of the cross-sectional propagation and lands closer to the historical slope than the Gaussian baseline, which systematically understates how strongly the stress reaches the most exposed names. The factor-level conditioning therefore carries through to a coherent, correctly signed response at the stock level.

5.2.3. Joint Conditioning

The conditions so far constrain a single factor. A more demanding test is to impose multiple extreme factor constraints at once, since the conditional cross-section is then governed not by one marginal but by the full comovement between the constrained factors and their joint spillover onto the remaining five. We consider two such joint regimes: a quality rally combined with a market drawdown and a volatility spike combined with the same market drawdown. Both are deep in the historical tail, with only $n = 9$ and $n = 27$ historical days respectively satisfying the constraints.



(a) Quality rally and market drawdown (quality $> 0.0032 \wedge$ market < -0.0247, $n = 9$).



(b) Volatility spike and market drawdown (volatility $> 0.0103 \wedge$ market < -0.0247, $n = 27$).

Figure 5.13: Per-factor conditional densities under two simultaneous factor constraints. Rows are the historical stress days, guidance Diffusion and the Gaussian baseline; columns are the seven factors; the dotted line is the unconditional baseline.

Figure 5.13 shows the per-factor densities for both joint regimes. The historical rows make the difficulty explicit: under a double constraint the comovement is genuinely complex, the unconstrained factors develop heavy shoulders, asymmetry and multiple modes rather than simply shifting, because so few days satisfy both conditions at once. The Gaussian baseline cannot follow this. Its conditional densities collapse to very thin, over-concentrated spikes, and on several factors they land in visibly wrong places, mode in the wrong tail or no overlap with the historical mass at all, precisely because a single fixed covariance forces an elliptical response that has no freedom to bend with the regime. Guidance, by contrast, reproduces the historical rows closely on both regimes: it keeps the heavier, multi-modal shape of the unconstrained factors and places its mass where the historical days do, rather than degenerating

to a spike. Joint conditioning is therefore the regime where the gap between our method and the elliptical baseline is widest, the guidance model captures a dependence-driven joint response that the Gaussian fundamentally cannot represent.

5.3. Temporal Generation

5.4. Downstream Risk Management

6

Ablation studies

The evaluation of Chapter 5 measures how well the proposed framework reproduces the stylised facts of the MSCI World panel. This chapter steps back and asks a more basic question about the approach itself: how well are diffusion models suited to financial factor data in the first place? Equity factor returns are an unusual target for a diffusion model — the panel is short by deep-learning standards, the marginals are heavy-tailed, and the cross-sectional signal is weak relative to the idiosyncratic noise. Any limitation observed in the main results could therefore stem either from the model's representational capacity or from the data regime it is trained in, and the two are easily confused. The ablations below are designed to separate them.

We probe two axes. Section 6.1 isolates the *dependence geometry* with a synthetic study in which the ground-truth covariance is prescribed exactly: it asks whether the model's tendency to wash out cross-factor correlation is intrinsic or an artefact of the near-isotropic spectrum of real factor returns. Section 6.2 isolates the *data regime*: it varies the length of the training panel to test whether the model's performance on MSCI is limited by sample size and whether training has in fact converged. Together they bound the question from both sides — what the model can represent given clean signal, and what it can achieve given the data actually available.

6.1. Diffusion Model on Learning covariance

We study which covariance structure can be learned better via diffusion models.

We start from the empirical correlation matrix C of the MSCI World factor panel and construct three ground-truth regimes that share the same eigenvector basis but differ in their eigenvalue spectrum:

- **synth**: the historical correlation C used as is;
- **amplified**: $C_\beta = \beta C + (1 - \beta)I$, which scales every off-diagonal entry by β while leaving the diagonal at one. We take β as large as positive-definiteness allows, $\beta = 0.99/(1 - \lambda_{\min})$, so the pairwise correlations are uniformly strengthened;
- **anisotropic**: the eigenvalues of C are raised to the power $p = 5$ and renormalised to preserve their mean (eigenvectors unchanged), then the matrix is rescaled back to unit diagonal. This stretches the spectrum, making the leading eigenvalue larger and the trailing.

To test how well a diffusion model can learn cross-factor dependence, we turn each of these three correlation matrices into a dataset and fit a separate model to it. For each regime we draw a sample of the same size as the real training panel from a multivariate Gaussian with the prescribed correlation, so that all three datasets share identical standard-normal marginals and differ only in their dependence geometry. Any difference in how well the model performs can therefore be attributed to the correlation structure alone. We then fit a separate diffusion model to each dataset, sample 4096 factor vectors from it, and ask how faithfully the generated samples reproduce the prescribed correlation. We measure this with three statistics that compare the empirical correlation matrix of the generated samples against that of the ground-truth draw: the Frobenius norm of their difference, the mean absolute error (MAE) of

the off-diagonal entries, and the Pearson correlation between the upper-triangular entries of the two matrices, which captures how well the pattern of pairwise dependence is preserved. We also compare their eigenvalue spectra directly (Figure 6.1), which shows where along the spectrum the model gains or loses fidelity.

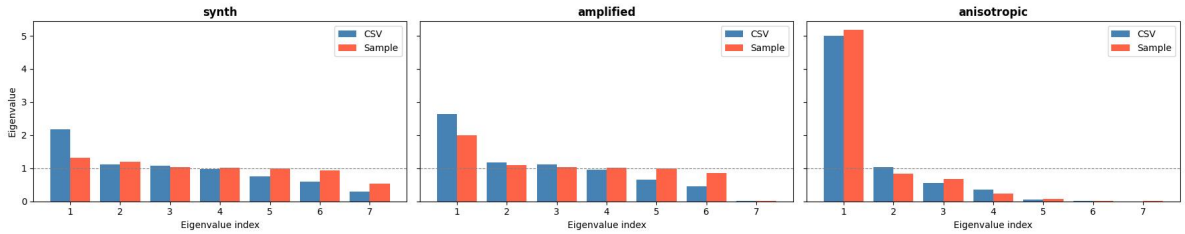


Figure 6.1: Eigenvalue spectra of the correlation matrices for the three regimes, comparing the ground truth (blue) with the diffusion-model samples (red). Starting from the empirical factor covariance, we construct three datasets whose correlation structure is progressively more anisotropic, that is with an increasingly uneven eigenvalue spectrum, from synth to anisotropic. The blue bars show these prescribed ground-truth spectra, and the dashed line marks the isotropic value of one. The red bars show that the model reproduces the ground-truth spectrum well in the anisotropic regime but flattens it toward one in the near-isotropic synth regime.

Table 6.1: Recovery of the cross-factor correlation structure by the diffusion model across the three synthetic regimes, ordered by increasing spectral anisotropy. Lower is better for the Frobenius norm and off-diagonal MAE; higher is better for the upper-triangular Pearson r.

Regime	Frobenius ↓	off-diag MAE ↓	upper-tri Pearson r ↑
synth	1.2528	0.1377	0.4934
amplified	1.2006	0.1454	0.8072
anisotropic	**0.5214**	**0.0607**	**0.9938**

Table 6.1 shows that recovery depends on the geometry of the spectrum, but not in the naive way. The synth regime, which uses the real factor correlation, is recovered poorly: the Frobenius distance to the ground truth is 1.25 and the upper-triangular Pearson correlation only 0.49. Simply strengthening every pairwise correlation, the amplified regime, does not fix this: the Frobenius distance (1.20) and the off-diagonal MAE (0.145 against 0.138) are essentially unchanged, and only the pattern of dependence aligns somewhat better (Pearson 0.81). It is only when the data is made genuinely anisotropic that recovery improves sharply, with the Frobenius distance more than halving to 0.52, the off-diagonal MAE dropping to 0.061, and the Pearson correlation reaching 0.99. The heatmaps in Figure 6.2 tell the same story visually: in both the synth and amplified regimes the generated off-diagonal entries are compressed toward zero and stay well below the ground-truth values, whereas in the anisotropic regime the generated matrix reproduces both the sign and the magnitude of every pairwise correlation.

The result shows that our diffusion model recovers the joint dependence structure better, when the covariance spectrum is more anisotropic. The mechanism is consistent with the isotropic mean-squared-error objective of the denoiser: when the spectrum is flat, the weak off-diagonal signal is of the same order as the per-coordinate noise the objective is indifferent to, and the generated covariance collapses toward the identity; when a few large eigenvalues dominate, the dependence becomes a high-variance direction that the objective is forced to fit, and the off-diagonal structure is recovered almost exactly.

6.2. Data convergence on MSCI

This ablation study investigates whether the model's performance can be improved by providing more data. If more history kept improving the generated statistics, the shortfall would be a sample-size limitation.

We retrain the model from scratch on training windows of increasing length and track how the evaluation statistics respond. Concretely we run an expanding-window walk-forward over the full 25 years: the training window ends in year and the test year is the immediately following year $Y+1$. Sweeping Y gives 24 folds whose training length grows from a single year to 24 years each retrained to convergence from

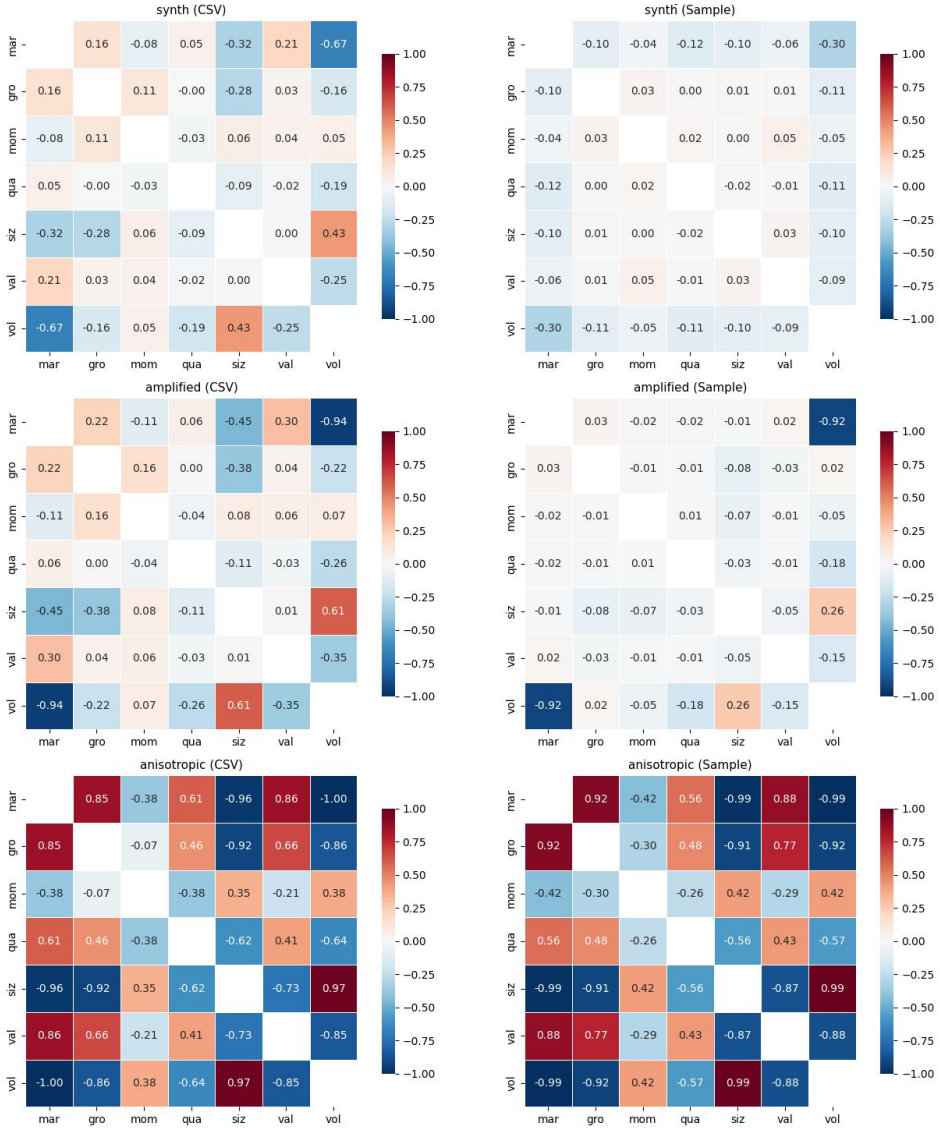


Figure 6.2: Empirical correlation matrices for the three synthetic regimes (rows: synth, amplified, anisotropic). The left column is the ground-truth draw and the right column the diffusion-model samples. In the synth regime, which uses the real factor correlation, the generated off-diagonal structure (right) is visibly washed out toward zero. Strengthening every pairwise correlation in the amplified regime does not restore it: the generated entries are still far weaker than the ground-truth values. Only in the anisotropic regime does the generated matrix recover the ground truth, with the two columns nearly indistinguishable.

random initialisation under an identical configuration.For each fold we compare the 4096 generated vectors against that fold's OOS year on five statistics: the mean-absolute error (MAE) across the seven factors of the first four marginal moments (mean, variance, skewness, kurtosis) and the relative Frobenius distance between the generated and OOS covariance matrices.

Figure 6.3 and Table 6.2 reveal three distinct behaviours across the five statistics. The mean MAE is already small at the shortest windows and shows no systematic trend with training length, indicating that the first moment has effectively converged regardless of sample size. The variance MAE is relatively stable across folds, with pronounced spikes confined to crisis test years (the 2008 and 2020 test years), suggesting that more data helps the second moment only modestly and that its largest errors are driven by the difficulty of the test year rather than by a shortage of training data. The remaining three statistics(skewness, kurtosis, and the covariance Frobenius distance) behave erratically: they are highly noisy from fold to fold and, if anything, fail to improve or even worsen as more data is added. These higher-order and cross-sectional properties are evidently the hardest for the model to learn, and additional history does not resolve them.

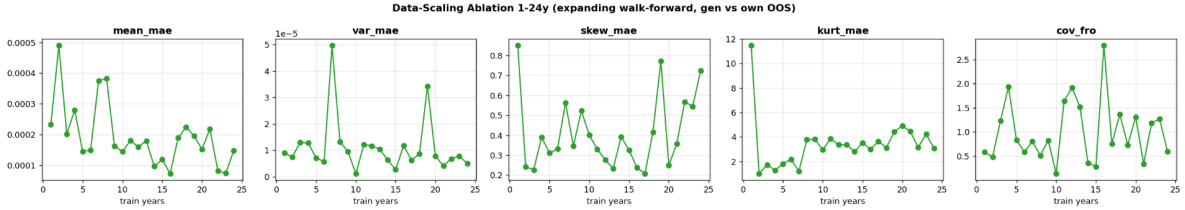


Figure 6.3: Generation quality versus training-set length under the expanding-window walk-forward. Each panel shows one evaluation statistic against the number of training years, lower is better.

Table 6.2: Data-scaling sweep: generation statistics by training-window length, each fold evaluated against its own test year. n is the number of training days. Mean and variance MAE are reported in scaled units (columns headed $\times 10^{-4}$ and $\times 10^{-5}$); skewness MAE, kurtosis MAE, and the relative covariance Frobenius distance are unscaled. Lower is better throughout.

Train yrs	test yr	n	Mean MAE ($\times 10^{-4}$)	Var MAE ($\times 10^{-5}$)	Skew MAE	Kurt MAE	Cov Frob
1	2002	261	2.33	0.90	0.851	11.484	0.581
2	2003	522	4.91	0.74	0.242	1.001	0.483
3	2004	783	2.02	1.30	0.227	1.740	1.235
4	2005	1045	2.80	1.27	0.389	1.282	1.934
5	2006	1305	1.45	0.71	0.311	1.804	0.834
6	2007	1565	1.50	0.57	0.332	2.189	0.582
7	2008	1826	3.76	4.97	0.563	1.207	0.807
8	2009	2088	3.83	1.32	0.347	3.791	0.510
9	2010	2349	1.63	0.95	0.523	3.811	0.826
10	2011	2610	1.45	0.11	0.401	2.967	0.135
11	2012	2870	1.81	1.21	0.330	3.867	1.643
12	2013	3131	1.60	1.17	0.277	3.367	1.914
13	2014	3392	1.80	1.04	0.233	3.372	1.516
14	2015	3653	0.97	0.63	0.392	2.801	0.356
15	2016	3914	1.20	0.26	0.325	3.544	0.284
16	2017	4175	0.73	1.17	0.237	2.991	2.793
17	2018	4435	1.89	0.62	0.207	3.646	0.758
18	2019	4696	2.25	0.87	0.416	3.129	1.364
19	2020	4957	1.96	3.42	0.773	4.442	0.728
20	2021	5219	1.53	0.78	0.250	4.911	1.312
21	2022	5480	2.19	0.40	0.358	4.475	0.340
22	2023	5740	0.82	0.68	0.568	3.154	1.188
23	2024	6000	0.75	0.78	0.545	4.232	1.267
24	2025	6262	1.48	0.50	0.724	3.063	0.596

In summary, on our panel the mean has essentially converged, enlarging the training set improves the variance only to a limited extent, and the behaviour of the remaining three statistics is inconclusive, they are too noisy from fold to fold to attribute any clear gain to a larger training set.

References

[1] Juan Miguel Lopez Alcaraz and Nils Strodthoff. "Diffusion-based Time Series Imputation and Forecasting with Structured State Space Models". In: *Transactions on Machine Learning Research*. 2022.

[2] Giovanni Barone-Adesi, Kostas Giannopoulos, and Les Vosper. "VaR without correlations for portfolios of derivative securities". In: *Journal of Futures Markets: Futures, Options, and Other Derivative Assets* 19.5 (1999), pp. 583–602.

[3] Minshuo Chen et al. "Diffusion Factor Models: Generating High-Dimensional Returns with Factor Structure". In: *arXiv preprint arXiv:2504.06566* (2025).

[4] Jooyoung Choi et al. "ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 14347–14356.

[5] Hyungjin Chung et al. "Diffusion Posterior Sampling for General Noisy Inverse Problems". In: *The Eleventh International Conference on Learning Representations (ICLR)*. 2023.

[6] Rama Cont. "Empirical properties of asset returns: stylized facts and statistical issues". In: *Quantitative Finance* 1.2 (2001), pp. 223–236.

[7] Prafulla Dhariwal and Alexander Nichol. "Diffusion Models Beat GANs on Image Synthesis". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 8780–8794.

[8] Yitong Duan et al. "FactorVAE: A Probabilistic Dynamic Factor Model Based on Variational Autoencoder for Predicting Cross-Sectional Stock Returns". In: *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*. 2022, pp. 4468–4476.

[9] Paul Embrechts, Alexander McNeil, and Daniel Straumann. "Correlation and dependence in risk management: properties and pitfalls". In: *Risk Management: Value at Risk and Beyond*. Cambridge University Press, 2002, pp. 176–223.

[10] Eugene F Fama and Kenneth R French. "Common risk factors in the returns on stocks and bonds". In: *Journal of Financial Economics* 33.1 (1993), pp. 3–56.

[11] Eugene F Fama and James D MacBeth. "Risk, return, and equilibrium: Empirical tests". In: *Journal of Political Economy* 81.3 (1973), pp. 607–636.

[12] Xuefeng Gao, Mengying He, and Xuedong He. "Factor-Based Conditional Diffusion Model for Portfolio Optimization". In: *arXiv preprint arXiv:2509.22088* (2025).

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020, pp. 6840–6851.

[14] Jonathan Ho and Tim Salimans. "Classifier-Free Diffusion Guidance". In: *arXiv preprint arXiv:2207.12598* (2022).

[15] Xingchang Huang et al. "Blue Noise for Diffusion Models". In: *ACM SIGGRAPH 2024 Conference Papers*. 2024.

[16] J.P. Morgan and Reuters. *RiskMetrics – Technical Document*. Tech. rep. New York: J.P. Morgan/Reuters, 1996.

[17] Tero Karras et al. "Elucidating the Design Space of Diffusion-Based Generative Models". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 26549–26561.

[18] Zhifeng Kong et al. "DiffWave: A Versatile Diffusion Model for Audio Synthesis". In: *International Conference on Learning Representations*. 2021.

[19] Szymon Kubiak et al. "MacroVAE: Counterfactual Financial Scenario Generation via Macroeconomic Conditioning". In: *Proceedings of the 6th ACM International Conference on AI in Finance*. 2025, pp. 797–805. DOI: 10.1145/3768292.3770360.

[20] Andrew W Lo and A Craig MacKinlay. “Stock market prices do not follow random walks: Evidence from a simple specification test”. In: *The Review of Financial Studies* 1.1 (1988), pp. 41–66.

[21] Andreas Lugmayr et al. “RePaint: Inpainting Using Denoising Diffusion Probabilistic Models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 11496–11505.

[22] Gábor Lugosi and Shahar Mendelson. “Mean estimation and regression under heavy-tailed distributions: A survey”. In: *Foundations of Computational Mathematics* 19.5 (2019), pp. 1145–1190.

[23] Chenlin Meng et al. “SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations”. In: *International Conference on Learning Representations (ICLR)*. 2022.

[24] Alexander Quinn Nichol and Prafulla Dhariwal. “Improved Denoising Diffusion Probabilistic Models”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8162–8171.

[25] William Peebles and Saining Xie. “Scalable Diffusion Models with Transformers”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 4195–4205.

[26] Dimitris N Politis and Joseph P Romano. “The stationary bootstrap”. In: *Journal of the American Statistical Association* 89.428 (1994), pp. 1303–1313.

[27] Kashif Rasul et al. “Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8857–8868.

[28] Dario Shariatian, Umut Simsekli, and Alain Durmus. “Denoising Lévy Probabilistic Models”. In: *Advances in Neural Information Processing Systems*. 2024.

[29] William F Sharpe. “Capital asset prices: A theory of market equilibrium under conditions of risk”. In: *The Journal of Finance* 19.3 (1964), pp. 425–442.

[30] Jascha Sohl-Dickstein et al. “Deep Unsupervised Learning using Nonequilibrium Thermodynamics”. In: *Proceedings of the 32nd International Conference on Machine Learning*. Vol. 37. Proceedings of Machine Learning Research. PMLR, 2015, pp. 2256–2265.

[31] Jiaming Song et al. “Loss-Guided Diffusion Models for Plug-and-Play Controllable Generation”. In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, pp. 32483–32498.

[32] Yang Song and Stefano Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019, pp. 11918–11930.

[33] Yang Song and Stefano Ermon. “Improved Techniques for Training Score-Based Generative Models”. In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020, pp. 12438–12448.

[34] Yang Song et al. “Score-Based Generative Modeling through Stochastic Differential Equations”. In: *International Conference on Learning Representations*. 2021.

[35] Yusuke Tashiro et al. “CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation”. In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 24804–24816.

[36] Ling Yang et al. “Diffusion Models: A Comprehensive Survey of Methods and Applications”. In: *ACM Computing Surveys* (2023).

A

Experimental Details

A.1. Sample Size Determination

To determine a stable sample size for evaluation, we generate 4096 samples from each method and slice sample sizes as $\{32, 64, 128, 256, 512, 1024, 2048, 4096\}$. We select the size where the distance to the training set mean becomes stable.

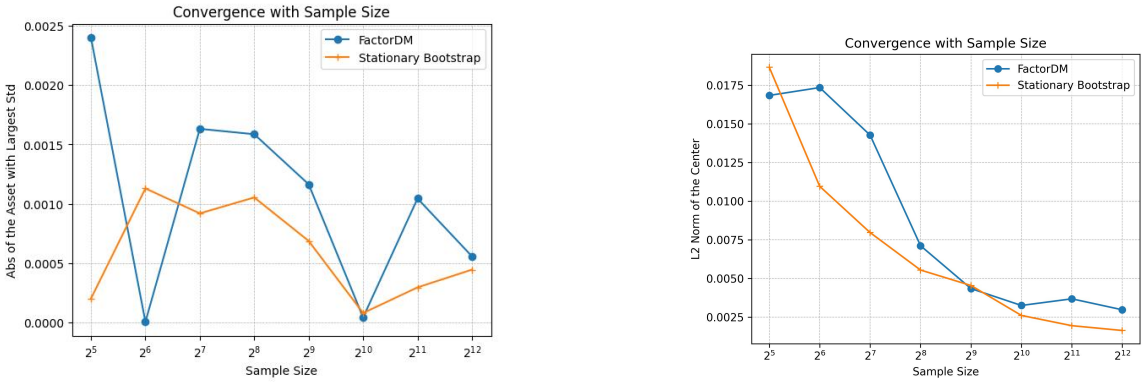


Figure A.1: Distance to training set as a function of sample size: all stocks (left) and highest-variance stock (right).

Based on these results, we use sample size = 2048 for both methods to balance stability and efficiency.

A.2. Downstream Tasks

Generated scenarios can be applied to two standard downstream tasks in quantitative finance: risk management and portfolio optimization.

A.2.1. Risk Management

For each method, portfolio returns are computed as $r_p = X \cdot w$, where $X \in \mathbb{R}^{M \times N}$ is the matrix of generated cross-sectional returns and $w \in \mathbb{R}^N$ are fixed portfolio weights. Risk metrics are then estimated empirically from the resulting return distribution. We report four metrics:

- **VaR**$_{5\%}$: Value at Risk at the 5% left tail, $\text{VaR}_\alpha = \inf\{l : P(r_p \leq -l) \leq \alpha\}$.
- **ES**$_{5\%}$: Expected Shortfall, $\text{ES}_\alpha = \mathbb{E}[r_p \mid r_p \leq \text{VaR}_\alpha]$.
- **Vol**: standard deviation of portfolio returns σ_{r_p}.
- **MaxLoss**: worst single-scenario portfolio return $\min_i r_{p,i}$.

We evaluate two portfolio strategies: an **equal-weight** portfolio ($w_i = 1/N$) and a **minimum-variance** portfolio ($\min_w \; w^\top \Sigma w$, s.t. $\sum_i w_i = 1$, $w_i \geq 0$, with at most 20 non-zero weights). Results are compared

against the historical ground truth (GT).

Table A.1: Cross-sectional risk metrics for equal-weight and minimum-variance portfolios.

Portfolio	Method	$VaR_{5\%}$	$ES_{5\%}$	Vol	MaxLoss
Equal-Weight	Historical (GT)	-0.0120	-0.0168	0.0072	-0.0385
	FactorDM	-0.0114	-0.0164	0.0066	-0.0350
	Stationary Bootstrap	-0.0109	-0.0158	0.0070	-0.0367
Min-Variance	Historical (GT)	-0.0080	-0.0120	0.0054	-0.0349
	FactorDM	-0.0072	-0.0107	0.0047	-0.0321
	Stationary Bootstrap	-0.0074	-0.0112	0.0052	-0.0237

For the Stationary Bootstrap, which produces multi-day trajectories, we additionally evaluate temporal risk over horizons $T \in \{5, 10, 20\}$ days by computing cumulative portfolio returns per scenario.

Table A.2: Temporal risk metrics for the Stationary Bootstrap at varying horizons.

Portfolio	T (days)	Method	$VaR_{5\%}$	$ES_{5\%}$	Vol
Equal-Weight	5	Historical (GT)	-0.0232	-0.0348	0.0155
	5	Stationary Bootstrap	-0.0239	-0.0315	0.0154
	10	Historical (GT)	-0.0285	-0.0441	0.0207
	10	Stationary Bootstrap	-0.0311	-0.0410	0.0214
	20	Historical (GT)	-0.0335	-0.0529	0.0271
	20	Stationary Bootstrap	-0.0431	-0.0526	0.0293
Min-Variance	5	Historical (GT)	-0.0189	-0.0258	0.0117
	5	Stationary Bootstrap	-0.0172	-0.0233	0.0119
	10	Historical (GT)	-0.0239	-0.0345	0.0165
	10	Stationary Bootstrap	-0.0228	-0.0313	0.0169
	20	Historical (GT)	-0.0284	-0.0387	0.0228
	20	Stationary Bootstrap	-0.0320	-0.0430	0.0238

A.2.2. Portfolio Optimization

Each method's generated scenarios are used to estimate the mean return vector μ and covariance matrix Σ, which are then fed into a long-only max-Sharpe optimisation:

$$\max_{w} \frac{\mu^\top w}{\sqrt{w^\top \Sigma w}} \quad \text{s.t.} \quad \sum_i w_i = 1,\ w_i \geq 0.$$

The resulting portfolio is evaluated on historical (GT) data. For reference, the GT efficient frontier is computed by minimising volatility subject to a target return constraint, sweeping from the minimum-variance return to the maximum single-asset return.

A.3. Additional Conditional Generation Results

This section collects the full diagnostic suite for the conditional experiments of Section 5.2, covering the volatility stress (volatility < -0.0096) and the two momentum regimes (crash, momentum < -0.0083; rally, momentum > 0.0054). For each regime we report the per-factor conditional densities, factor correlation matrices, joint factor responses, and global-geometry (PCA / scatter) diagnostics, of which only a subset appears in the main text. The momentum crash is the one regime where the elliptical baseline edges out guidance on dependence: the whole-matrix relative mean-absolute correlation error against the historical stress days is 0.1909 for guidance, 0.1704 for the Gaussian baseline and 0.1510 for

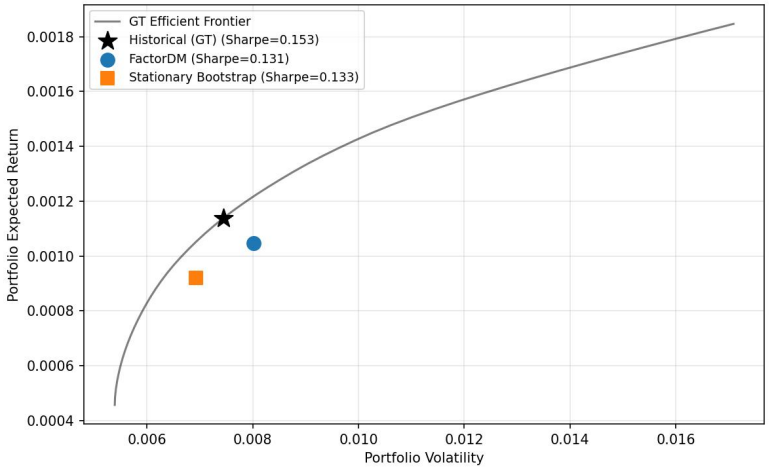


Figure A.2: Max-Sharpe portfolios evaluated on GT data, with the GT efficient frontier as reference.

the unconditional cross-section, reflecting the near-Gaussian shape of the momentum factor. Figures below are grouped by regime (volatility stress, then crash, then rally); captions identify each panel.

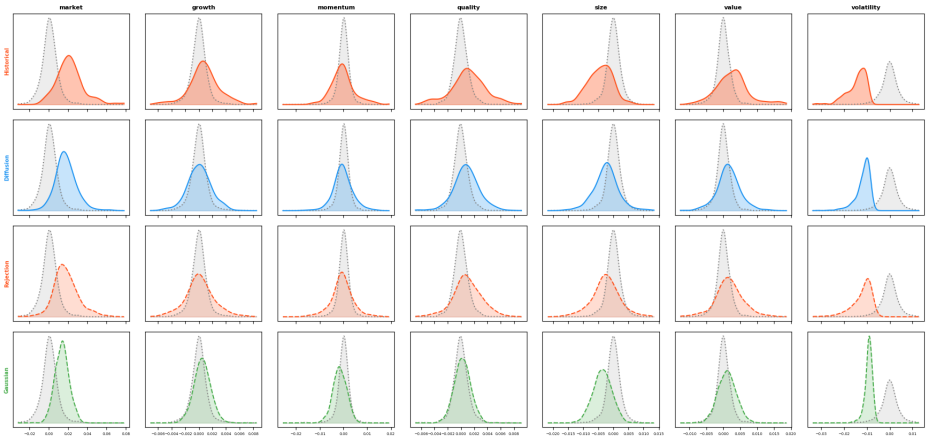


Figure A.3: Volatility stress: per-factor conditional densities for the historical stress days, guidance, soft rejection and the Gaussian baseline (rows), against the unconditional baseline (dotted), one column per factor.

Figure A.4: Volatility stress: factor correlation matrices for the unconditional, soft-rejection, guidance and Gaussian samples and for the historical stress days.

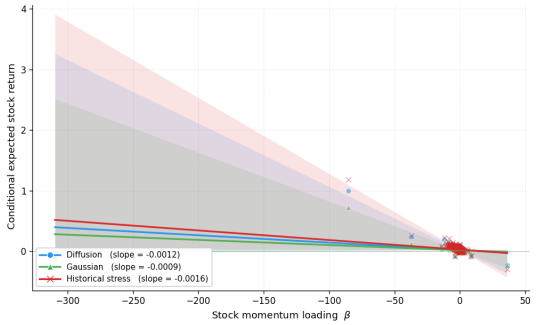


Figure A.5: Volatility stress: conditional expected stock return against the stock's momentum loading β, for guidance, the Gaussian baseline and the historical stress days.

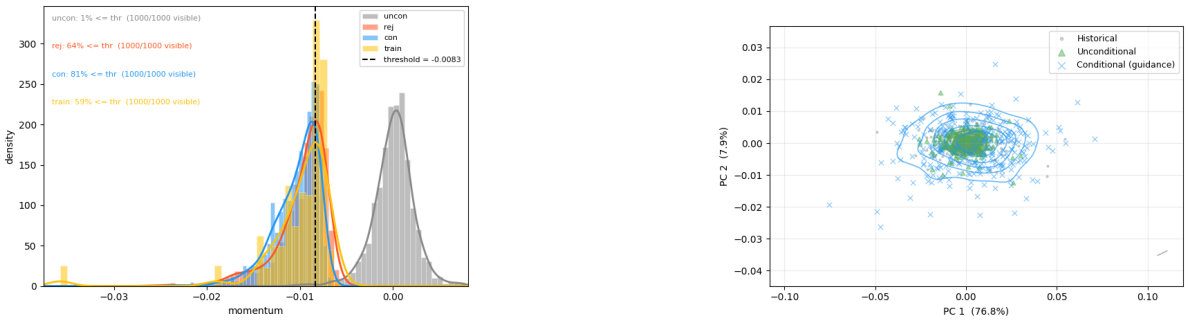


Figure A.6: Momentum crash: targeted-factor marginal under the condition (left) and PCA projection of the conditional and unconditional scenarios (right).

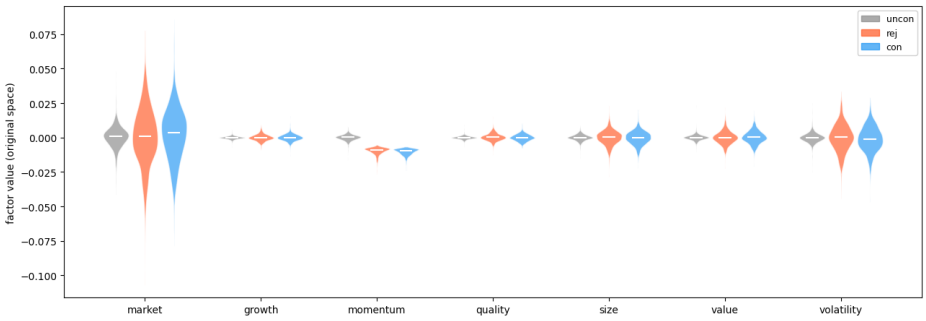


Figure A.7: Momentum crash: joint response of all seven factors for the unconditional, soft-rejection and guidance samples.

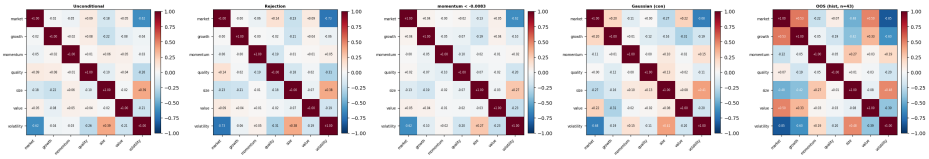


Figure A.8: Momentum crash: factor correlation matrices for the unconditional, soft-rejection, guidance and Gaussian samples and for the historical stress days.

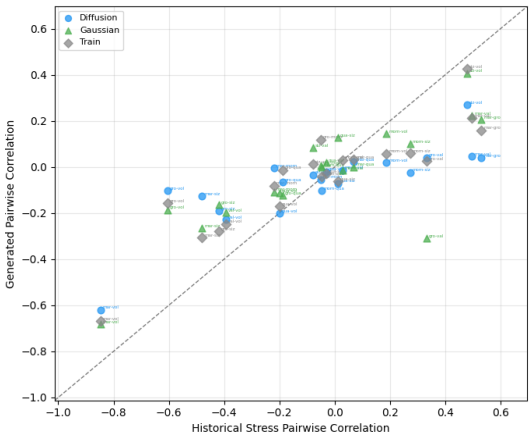


Figure A.9: Momentum crash: pairwise factor correlations of each method against the historical-stress correlations.

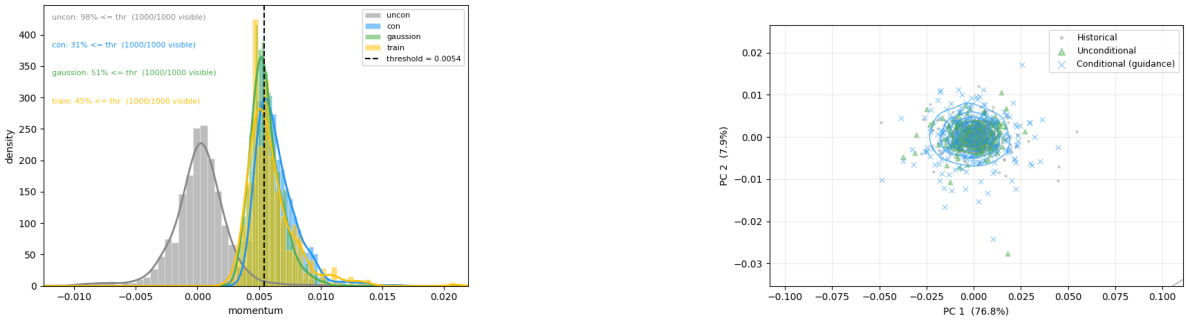


Figure A.10: Momentum rally: targeted-factor marginal under the condition (left) and PCA projection of the conditional and unconditional scenarios (right).

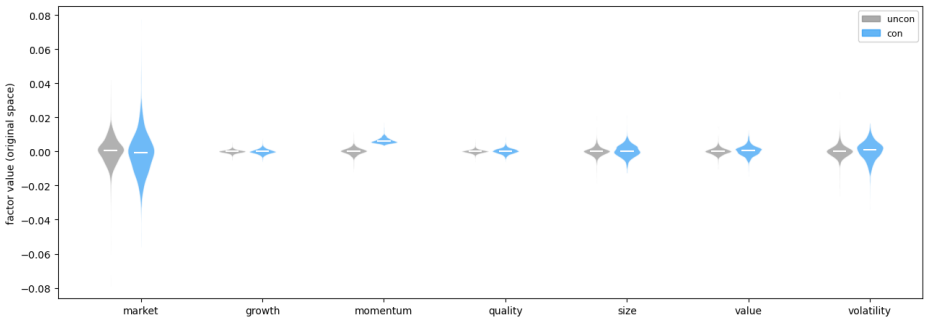


Figure A.11: Momentum rally: joint response of all seven factors for the unconditional, soft-rejection and guidance samples.

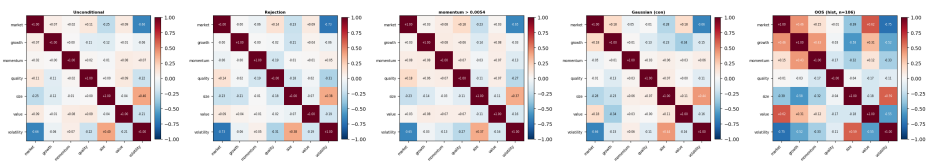


Figure A.12: Momentum rally: factor correlation matrices for the unconditional, soft-rejection, guidance and Gaussian samples and for the historical stress days.

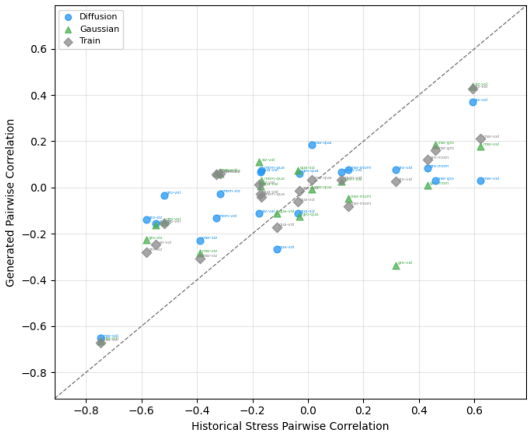


Figure A.13: Momentum rally: pairwise factor correlations of each method against the historical-stress correlations.

B

Notation

Symbols introduced in Chapter 3, grouped by topic.

Symbol	Meaning
Indexing and dimensions	
i	Stock index, $1 \le i \le N$
N	Number of stocks in the cross-section
K	Number of factors (= input dimension of the diffusion model)
T, t	Number of dates / date index in §3.1; **diffusion steps / step index** from §3.2 onwards
B	Minibatch size
M, K_z	Outer / inner Monte Carlo counts (`mc_outer`, `mc_inner`)
s	Synthesis-time sample subscript (§3.1); guidance strength (§3.2.2)
Factor model (§3.1)	
$R_{i,t}$	Stock i return at date t
R_t^f	Risk-free rate
$\mathbf{F}_t \in \mathbb{R}^K$	Factor return vector at date t, eq. (3.3)
a_i	Per-stock intercept (Jensen's alpha), eq. (3.2)
$\boldsymbol{\beta}_i \in \mathbb{R}^K$	Per-stock factor loadings, eq. (3.2)
$\varepsilon_{i,t}$	Per-stock idiosyncratic residual, eq. (3.2)
σ_i, ν_i	Student-t scale and degrees of freedom of stock i, eq. (3.4)
$\mathbf{b}_{i,t-1}$	Lagged firm characteristics of stock i, eq. (3.3)
$u_{i,t}$	Cross-sectional regression intercept (= market factor), eq. (3.3)
Diffusion forward process (§3.2.1)	
$\mathbf{x}_0$, $\mathbf{x}_t \in \mathbb{R}^K$	Clean / noised factor vector at diffusion step t, eq. (3.6)
$\alpha \in (0, 2]$	Lévy stability index ($\alpha = 2$ recovers Gaussian/DDPM)
$S(\alpha, 0)$	Symmetric α-stable distribution
γ_t, σ_t, $\bar{\gamma}_t$, $\bar{\sigma}_t$	Single-step / cumulative DLPM schedule coefficients, eq. (3.7)
α_t, β_t	Cosine schedule scalars ($\beta_t = 1 - \alpha_t$, DDPM convention)
$A_t \sim S(\alpha/2, 1)$	Positive subordinator (per factor), eq. (3.8)
$\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$	Gaussian draw
$\boldsymbol{\varepsilon}_t = \sqrt{A_t}\mathbf{L}\mathbf{z}_t$	Forward-step noise increment, eq. (3.10)
$\mathbf{C}$, $\mathbf{L}$	Empirical factor correlation and its Cholesky factor, $\mathbf{C} = \mathbf{L}\mathbf{L}^\top$
Σ_t	Accumulated forward variance given $A_{1:t}$, eq. (3.9)
Diffusion reverse process and conditional sampling (§3.2.2)	
$\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t)$	Learned noise-prediction network, eq. (3.13)
θ	Denoiser parameters
Γ_t	Posterior contraction factor, eq. (3.12)
$\boldsymbol{\mu}_t$	Reverse-step posterior mean, eq. (3.12)

Symbol	Meaning
$\hat{\mathbf{x}}_0$	Tweedie estimate of $\mathbf{x}_0$, eq. (3.15)
$\mathbf{c}$, $\mathcal{L}(\mathbf{x}_0, \mathbf{c})$	condition and condition loss eq. (3.14)
w_t, SNR_t, p	SNR-based decay weight, ratio and exponent, eq. (3.17)