

GENERATIVE LEARNING FOR FINANCIAL TIME SERIES WITH IRREGULAR AND SCALE-INVARIANT PATTERNS

Anonymous authors

Paper under double-blind review

ABSTRACT

Limited data availability poses a major obstacle in training deep learning models for financial applications. Synthesizing financial time series to augment real-world data is challenging due to the *irregular and scale-invariant* patterns uniquely associated with financial time series - temporal dynamics that repeat with varying duration and magnitude. Such dynamics cannot be captured by existing approaches, which often assume regularity and uniformity in the underlying data. We develop a novel generative framework called *FTS-Diffusion* to model irregular and scale-invariant patterns that consists of three modules. First, we develop a scale-invariant pattern recognition algorithm to extract recurring patterns that vary in duration and magnitude. Second, we construct a diffusion-based generative network to synthesize segments of patterns. Third, we model the temporal transition of patterns in order to aggregate the generated segments. Extensive experiments show that FTS-Diffusion generates synthetic financial time series highly resembling observed data, outperforming state-of-the-art alternatives. Two downstream experiments demonstrate that augmenting real-world data with synthetic data generated by FTS-Diffusion reduces the error of stock market prediction by up to 17.9%. To the best of our knowledge, this is the first work on generating intricate time series with irregular and scale-invariant patterns, addressing data limitation issues in finance.

1 INTRODUCTION

Researchers in financial economics have demonstrated intriguing potential for deep learning to solve complex problems in financial settings (Qin et al., 2017; Xu & Cohen, 2018; Wu et al., 2020; Manzo & Qiao, 2020; Huang & Li, 2021). However, a dearth of data and the low signal-to-noise ratio nature of financial data pose major obstacles that hinder the further development of deep learning in finance. Unlike the sciences, finance researchers cannot run experiments to obtain more data, so financial time series are limited by their existing history. Additionally, price and return data are subject to high levels of noise, making it even more challenging to extract useful information from a limited dataset. Deep learning models trained on insufficient data are prone to overfitting and cannot be expected to perform reliably on unseen data.

To alleviate data scarcity, data augmentation techniques can be employed. Generative models that capture the properties of the underlying data-generating process would produce synthetic data that resemble observed data. Recently, deep generative modeling, especially generative adversarial networks (GAN) (Goodfellow et al., 2014) and diffusion models (Ho et al., 2020), has made remarkable progress in multiple domains including image synthesis, reinforcement learning, and anomaly detection¹. They have also been applied to time series settings such as medical records, audio synthesis, power systems, and networked systems². Despite these advances, modeling financial time series poses unique challenges that complicate the task and render existing models ineffective.

The time series studied in the extant literature of deep generative learning tend to exhibit some regularity. Patterns identified in these data appear at fixed or predictable increments in calendar

¹Papers on images include Zhu et al. (2017), Turkoglu et al. (2019), Dhariwal & Nichol (2021), and Rombach et al. (2022). Reinforcement learning work includes Yu et al. (2017), Janner et al. (2022), and Eysenbach et al. (2022). Anomaly detection studies include Zenati et al. (2018) and Akcay et al. (2019).

²Medical: Esteban et al. (2017), Che et al. (2017), and etc. Audio: Kong et al. (2021), Leng et al. (2022), and etc. Power systems: Zhang et al. (2018) and Chen et al. (2018). Networked systems: Lin et al. (2020).

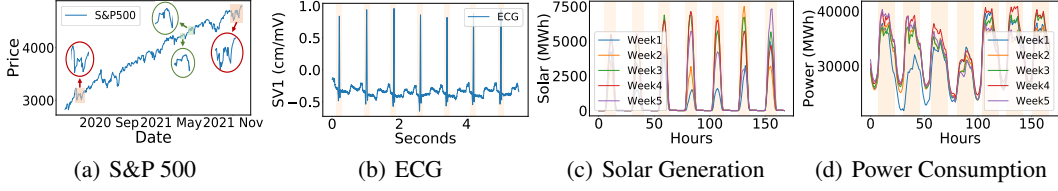


Figure 1: Time-series data of (a) S&P 500 price (finance), (b) ECG (medical), (c) solar generation (renewable energy), and (d) power consumption (smart grid). Unlike other time-series data with explicit patterns, financial time series expresses complex patterns that are irregular and scale-invariant.

time (e.g., heartbeats in ECG). Time series data that contain such *regular* patterns are amenable to modeling, as they allow the extraction of highly correlated features from similar repeating patterns. Although conceptually straightforward, identifying recurring patterns in financial time series proves difficult due to a lack of regularity. Instead, financial time series appear to contain more subtle patterns that repeat themselves with varying duration and magnitude, a quality we refer to as *scale-invariance*. *Irregularity* and *scale-invariance* are hallmarks of financial time series that complicate their modeling and the synthesis of additional data. We illustrate these two properties in Fig. 1 by comparing the S&P 500 Index, a broad basket of U.S. stocks, to several regular series. The three regular series exhibit clear and consistent patterns that align with calendar time. In contrast, we do not observe neat patterns that adhere to a fixed frequency for the S&P 500. Instead, we can observe similar patterns (red and green circles) that exhibit scale-invariance. These patterns keep their basic shape but are shifted or stretched compared to each other³. The unique properties of financial time series make data synthesis a significantly more challenging task compared to that of well-behaved data. Effective time series data generation considering irregularity and scale-invariance remains largely an open problem.

To address this problem, we deconstruct financial time series generation into a three-prong process: (i) *pattern recognition* to identify irregular and scale-invariant patterns, (ii) *generation* to synthesize segments of patterns, and (iii) *evolution* to connect the generated segments into a complete time series. We propose a new generative framework, *FTS-Diffusion* to accomplish the *pattern recognition-generation-evolution* process. We find that FTS-Diffusion is capable of generating synthetic financial time series that closely resemble observed data. We make the following contributions:

▷ We identify and define two properties of financial time series: irregularity and scale-invariance (see Sec. 3). We present a novel FTS-Diffusion framework to model time series data exhibiting these properties. To the best of our knowledge, this is the first framework capable of generating challenging time series data that contain irregularity and scale-invariance. FTS-Diffusion may also be applied to other domains with data exhibiting similar properties.

▷ The unique architecture of FTS-Diffusion is designed to handle irregularity and scale-invariance. There are three modules. The pattern recognition module is based on a new scale-invariant subsequence clustering (SISC) algorithm (Sec. 4.1). By incorporating dynamic time warping (DTW), SISC is able to accurately identify and separate irregular and scale-invariance patterns. The generation module consists of a diffusion-based network to synthesize the scale-invariant segments conditional on the patterns learned by SISC (Sec. 4.2). The evolution module is made up of a pattern transition network that produces the temporal evolution of consecutive patterns, capturing the dynamic relationship among them (Sec. 4.3).

▷ We demonstrate the effectiveness of FTS-Diffusion in capturing real-world financial data and we illustrate the value of the generated data for downstream applications (Sec. 5). Patterns identified by FTS-Diffusion can be cross-verified with financial domain knowledge (Lo et al., 2000), and experimental results from three real-world datasets show that FTS-Diffusion generates the most realistic financial time series among several alternative models. We explore the usage of the generated data for the downstream task of predicting stock prices. Augmenting limited real-world data with synthetic samples from FTS-Diffusion reduces the predictive error by up to 17.9% across the datasets. These results shed light on the capability of FTS-Diffusion to improve the accuracy and reliability of deep learning models in financial applications.

³For more details, Appendix A provides an in-depth discussion on the properties of financial time series.

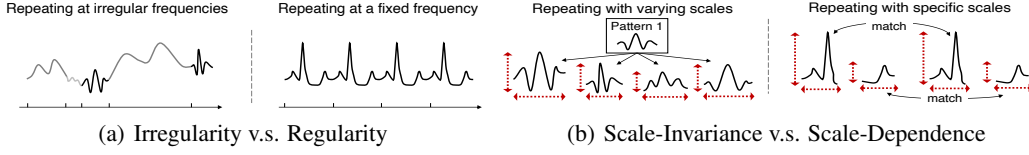


Figure 2: (a) Irregular patterns with indeterminate recurrence intervals vs. regular patterns repeated at a fixed frequency. (b) Scale-invariant patterns showing similarity after proper duration/magnitude scaling vs. scale-dependent patterns showing similarity given specific duration or magnitude.

2 RELATED WORK

Advances in deep generative modeling have shown promise to generate time series data in various problem domains, particularly using VAEs-, GANs-, and diffusion-based models. We discuss the most relevant works in this section.

TimeVAE (Desai et al., 2021) introduces a VAE-based framework to model the trend and seasonality in time series. RCGAN (Esteban et al., 2017) and MV-GAN (Brophy, 2020) use GANs for learning medical records. Several GAN variants are employed to model the time series in power systems (Zhang et al., 2018; Chen et al., 2018). TimeGAN (Yoon et al., 2019) develops a general framework for embedding time-series data into a latent space with an autoencoder network and subsequently learning the latent representation with GANs. QuantGAN (Wiese et al., 2020) provides a GAN-based network to capture long-range dependencies in financial time series under the volatility-innovation decomposition. CSDI (Tashiro et al., 2021) proposes a score-based diffusion model whose unconditional variant can be used for time series generation. DiffWave (Kong et al., 2021) and BinauralGrad (Leng et al., 2022) generate waveform time series with diffusion models.

The above approaches can model time series with regular patterns but struggle with more complex series characterized by irregularity and scale-invariance, central features in financial time series. The identification of latent patterns in financial time series is challenging, and it is difficult for a generative model without auxiliary information to distinguish between these diverse distributions. In our study, we decompose the financial time series generation into a *pattern recognition-generation-evolution* process, enabling better modeling of the irregular and scale-invariant properties. In addition, diffusion probabilistic models have been shown to achieve better quality and training stability than the classical GAN and VAE models (Dhariwal & Nichol, 2021; Wang et al., 2021). Hence, we design our generative model leveraging the denoising diffusion probabilistic model (DDPM) (Ho et al., 2020).

3 PROBLEM STATEMENT

3.1 UNIQUE CHARACTERISTICS OF FINANCIAL TIME SERIES

The irregular and scale-invariant patterns in financial time series are difficult for existing models to capture. Fig. 2 illustrates these repeating temporal dynamics with non-deterministic intervals and varying duration and magnitudes. Existing models that assume regularity and uniformity are unable to capture such patterns, because the typical approach of dividing time series into fixed-interval segments is likely to result in a snapshot of either a fraction of a pattern or a mixture of multiple patterns.

We propose a novel framework to model irregular and scale-invariant time series. A time series $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_M\}$ consists of M segments, $\mathbf{x}_m = \{x_{m,1}, \dots, x_{m,t_m}\}$. The length of the entire time series is $T = \sum_{m=1}^M t_m$. \mathbf{x}_m is sampled from a conditional distribution $f(\cdot|p, \alpha, \beta)$ dependent on the pattern $p \in \mathcal{P}$, whose duration is scaled by α and magnitude scaled by β . This way, \mathbf{x}_m will be statistically similar to its underlying pattern p while allowing for adjustments in duration and magnitude. To model the dynamics across patterns, we employ a Markov chain. Each tuple (p, α, β) is a state, and the state transition probabilities $Q(p_j, \alpha_j, \beta_j|p_i, \alpha_i, \beta_i)$ describe the stochastic transition from one pattern to the next. Our setup is reminiscent of applications of the Markov property in financial time series (Dueker, 1997; Bai & Wang, 2011; Somani et al., 2014). The novelty

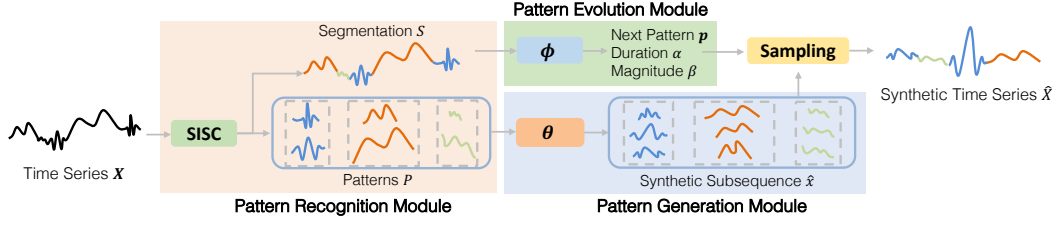


Figure 3: The architecture of our proposed FTS-Diffusion. The pattern recognition module identifies the scale-invariant patterns within the entire financial time series using our proposed SISC algorithm. Subsequently, the pattern generation module synthesizes the segments with a diffusion-based network conditioned on the patterns. Finally, the pattern evolution module connects the generated segments to construct a synthetic financial time series following the transition between consecutive patterns.

in our approach is that we use a Markov model to capture the transition of three specific aspects of the time series: pattern, duration, and magnitude, whereas existing work attempts to recover some unspecified latent properties of a time series.

3.2 PROBLEM STATEMENT

We seek to operationalize the structure laid out in Sec. 3.1. When faced with a time series, we have no knowledge of the segments $\{x_m\}_{m=1}^M$, the set of scale-invariant patterns \mathcal{P} , or the scaling factors α and β that transform a reference pattern into its more realistic counterpart. We also do not know the transition probabilities $Q(p_j, \alpha_j, \beta_j | p_i, \alpha_i, \beta_i)$. Our goal is to develop a data-driven framework to accomplish the following:

- (Pattern Recognition) identify the patterns and learn the recurrent structures \mathcal{P} , and group segments into clusters according to their corresponding patterns $p \in \mathcal{P}$;
- (Pattern Generation) learn the distribution $f(\cdot | p, \alpha, \beta)$, $\forall p \in \mathcal{P}$;
- (Pattern Evolution) learn the pattern transition probabilities $Q(p_j, \alpha_j, \beta_j | p_i, \alpha_i, \beta_i)$.

The three components above allow us to generate financial time series by (i) determining the allocation of patterns using the pattern transition probabilities, and (ii) generating each segment from the corresponding pattern with the appropriate duration and magnitude scaling factors. Our three-pronged framework dedicated to identifying and modeling the irregular and scale-invariant patterns observed in financial time series is the first of its kind in the literature.

4 OUR PROPOSED FTS-DIFFUSION FRAMEWORK

In this section, we present our FTS-Diffusion framework. Fig. 3 provides an illustration. FTS-Diffusion consists of three components: a pattern recognition module, a pattern generation module, and a pattern evolution module. Next, we introduce each module and how they work together.

4.1 PATTERN RECOGNITION: IDENTIFYING IRREGULAR AND SCALE-INVARIANT PATTERNS

We propose a novel Scale-Invariant Subsequence Clustering (SISC) algorithm to partition the entire financial time series into segments of variable lengths and group them into K distinct clusters. The segments within the same cluster exhibit similar shapes after proper scaling in duration and magnitude. The centroid of each cluster then represents a scale-invariant pattern in the financial time series.

The idea is similar to the traditional K-Means clustering (Hartigan & Wong, 1979), which primarily clusters segments of identical length and thus falls short in our context due to its inability to handle segments of varying lengths and magnitudes. Instead of dividing the time series into equal-length segments as is commonly done, we adaptively determine the optimal segment lengths through a simple yet effective greedy segmentation strategy. Specifically, as illustrated in Fig. 4(a), we compare

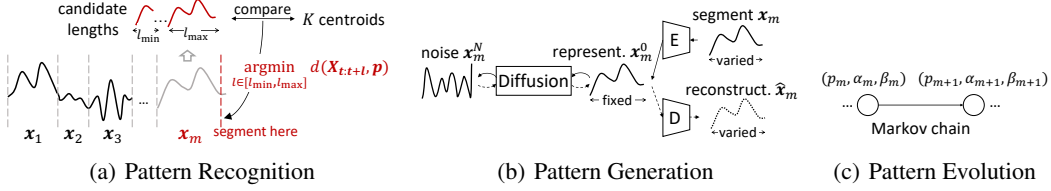


Figure 4: Key designs in each module of our FTS-Diffusion: (a) pattern recognition module: SISC greedy segmentation; (b) pattern generation module: the pattern-conditioned diffusion network paired with the scaling autoencoder; (c) pattern evolution module: Markov transition.

the segments of candidate lengths $l \in [l_{\min}, l_{\max}]$ with the cluster centroids at each evaluated position $t = \sum_{\tau=0}^{m-1} t_{\tau}$. The length that minimizes the distance to the nearest centroid is considered the optimal segmentation for the current segment $x_m = X_{t:t+l^*}$, i.e.,

$$l^* = \arg \min_{l \in [l_{\min}, l_{\max}]} d(X_{t:t+l}, \mathbf{p}), \forall \mathbf{p} \in \mathcal{P}. \quad (1)$$

The first key component in our design is a distance metric $d(\cdot, \cdot)$ that is robust to varying lengths and magnitudes and hence properly measures the difference between subsequences. Classical metrics, such as Euclidean distance, fail to provide accurate measurements due to their limitations in comparing variable-length sequences. In contrast, we employ dynamic time warping (DTW) to calculate the minimum cost across all pointwise alignments between two segments:

$$DTW(\mathbf{x}, \mathbf{y}) := \min_{A \in \mathcal{A}} \langle A, \Delta(\mathbf{x}, \mathbf{y}) \rangle, \quad (2)$$

where A denotes the alignment between two sequences in the set of all possible alignments \mathcal{A} , and $\Delta(\mathbf{x}, \mathbf{y}) = [\delta(x_i, y_j)]_{ij}$ is the pointwise distance matrix between two sequences \mathbf{x} and \mathbf{y} . It well-suits our purpose to identify similarities in segments with similar shapes but varying duration and magnitudes. Equipped with the DTW metric, we apply the greedy segmentation strategy from the start to the end of the time series. Upon completing the greedy segmentation across the entire time series, we proceed with the standard K-means clustering process. Each segment is assigned to the nearest centroid, then the centroids are updated based on current cluster assignments. This process iterates until cluster assignments stabilize or a pre-determined number of iterations is reached.

The second key component in our design is the initialization of the cluster centroids. The random initialization typically used in standard clustering methods often yields suboptimal clustering results. To alleviate this issue, we design a wise initialization that facilitates a more informed selection of initial centroids. Our initialization begins by randomly selecting one segment from all available segments of a pre-specified length to serve as the first centroid. Afterward, we choose the subsequent centroids from the remaining segments, with the selection weight being proportional to their distances to the closest centroid within the chosen set. This means that segments located farther from their nearest centroid have a higher probability of being selected as the next centroid. We repeat this process until K centroids have been initialized. This method ensures a diverse set of centroids spreading across the data space, which promotes an efficient start of our SISC clustering algorithm.

A pseudo-code of our SISC algorithm is provided in Appendix B.1. The computational complexity of SISC is $\mathcal{O}(TKl_{\max})$, which is linear to the length of the entire time series. In numerical experiments in Sec. 5, we set the minimum and maximum segment lengths as 10 and 21, respectively, focusing on the atom-like short-term patterns commonly observed in financial technical analysis (Lo et al., 2000). The number of clusters K is empirically determined by the elbow method (Thorndike, 1953).

We highlight that our SISC algorithm is the first in the literature designed to identify scale-invariant patterns. The learned patterns are cross-validated with the technical patterns in the financial literature (Lo et al., 2000) in Appendix B.2. To better verify the effectiveness of our proposed SISC algorithm, we conduct a thorough investigation using simulated time series data in Appendix B.3.

4.2 PATTERN GENERATION: LEARNING PATTERN-CONDITIONED TEMPORAL DYNAMICS

We develop a pattern generation module, θ in Fig. 3, to synthesize the segments of patterns. The goal is to generate new segments that mimic the temporal dynamics within the observed segments.

Considering the financial time series as a collection of scale-invariant patterns, the data-generating process can be interpreted as capturing the distribution of the reference patterns and transforming them with proper scales in duration and magnitude. Accordingly, we instantiate this data-generating process using two dedicated networks for the two tasks, as discussed below.

The first network is the pattern-conditioned diffusion network for simulating a stochastic process conditioned on the patterns - perturbing the pattern representation gradually by adding noise over N steps until it becomes pure noise (diffusion), and then recovering the pure noise backward to the original representation by gradually removing the noise over the same steps (denoising). The diffusion process is achieved by a pre-specified procedure of incrementally adding Gaussian noise step by step, while the denoising process is approximated by a neural network that learns the removing noise at each step, i.e., the denoising gradient. Approximating the stepwise denoising gradients is equivalent to learning the mapping from a latent Gaussian space to the pattern space. Consequently, given a Gaussian noise, we can generate a pattern representation. The continuous nature of the Gaussian space implies that we can sample an infinite amount of Gaussian noise and produce corresponding new pattern representations. We build our diffusion network based on the denoising diffusion probabilistic model (DDPM) (Ho et al., 2020). In detail, we apply the following diffusion process to corrupt the observed segments to pure noise:

$$\mathbf{x}^N = \mathbf{x}^0 + \sum_{i=0}^{N-1} \mathcal{N}(\mathbf{x}^{i+1}; \sqrt{1 - \beta}(\mathbf{x}^i - \mathbf{p}), \beta I), \quad (3)$$

where β represents the magnitude of the segments. Thereafter, we design a conditional denoising process that recovers the target temporal dynamics from a prior Gaussian noise conditioned on the reference patterns over N steps:

$$\mathbf{x}^0 = \mathbf{x}^N - \sum_{i=0}^{N-1} \epsilon_\theta(\mathbf{x}^{i+1}, i, \mathbf{p}), \quad (4)$$

where ϵ_θ is the neural network that learns the denoising gradient at each step. Note that the superscript i denotes the step in the diffusion and denoising process.

The second network is the scaling autoencoder (AE) for learning the transformation between variable-length segments \mathbf{x} and fixed-length representations \mathbf{x}^0 , after we capture the reference pattern representation using the pattern-conditioned diffusion network. The encoder of the scaling AE stretches the variable-length segments into fixed-length representations that align with the dimension of reference patterns. The decoder, on the other hand, is responsible for reconstructing the variable-length segments from the fixed-length representations.

The network structures of these two components are presented in Appendix C.2

We jointly train the pattern-conditioned diffusion network and the scaling AE using the standard supervised learning with the segments identified in Sec. 4.1 as training data. As illustrated in Fig. 4(b), the observed segments are encoded and perturbed to pure noise by the encoder in the scaling AE and the diffusion process in the pattern-conditioned diffusion network. The reverse direction, depicted by dashed-lined arrows, simulates the generation process that recovers and decodes the reconstructed segments from Gaussian noise through the denoising process in our diffusion network and the decoder in our scaling AE. During this process, we must ensure that (i) the diffusion and denoising gradients are consistent at each step, and (ii) the reconstruction successfully reproduces the observed segments. Therefore, the objective contains the reconstruction loss between the observed and reconstructed segments for the scaling AE and the unweighted variant of the variational lower bound (ELBO) (Ho et al., 2020) for the pattern-conditioned diffusion network:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}_m} [\|\mathbf{x}_m - \hat{\mathbf{x}}_m\|_2^2] + \mathbb{E}_{\mathbf{x}_m^0, i, \epsilon} [\|\epsilon^i - \epsilon_\theta(\mathbf{x}_m^{i+1}, i, \mathbf{p})\|_2^2], \quad (5)$$

where ϵ^i is the noise added in the corresponding diffusion process at step i .

4.3 PATTERN EVOLUTION: LEARNING THE TRANSITION BETWEEN CONSECUTIVE PATTERNS

As mentioned in Sec. 3.1, we model the transition states (encompassing patterns, lengths, and magnitudes) between consecutive generated segments using a Markov chain. Once the transition

states are determined, we obtain an evolution series of patterns, somehow addressing the irregularity in the financial time series. This ensures that the consecutive generated segments maintain the essential temporal correlations observed in real-world financial data. To capture the Markov-chain modeled temporal dynamics across patterns, we introduce a pattern evolution network ϕ , with the network structure in Appendix D.1 to learn the temporal evolution of the states between consecutive segments⁴. More specifically, the network learns the probability of the next pattern along with its corresponding length and magnitude, given the current state (because of the Markov property):

$$(\hat{p}_{m+1}, \hat{\alpha}_{m+1}, \hat{\beta}_{m+1}) = \phi(p_m, \alpha_m, \beta_m), \quad (6)$$

where $(\hat{p}_{m+1}, \hat{\alpha}_{m+1}, \hat{\beta}_{m+1})$ denotes the next pattern and its scales in length and magnitude.

The pattern evolution network is trained to optimize the following objective:

$$\mathcal{L}(\phi) = \mathbb{E}_{\mathbf{x}_m} [\ell_{CE}(p_{m+1}, \hat{p}_{m+1}) + \|\alpha_{m+1} - \hat{\alpha}_{m+1}\|_2^2 + \|\beta_{m+1} - \hat{\beta}_{m+1}\|_2^2], \quad (7)$$

where $\ell_{CE}(\cdot, \cdot)$ represents the cross-entropy.

4.4 PUTTING EVERYTHING TOGETHER: SYNTHESIZING ENTIRE FINANCIAL TIME SERIES

We regard patterns as the basic building blocks of generation. Accordingly, FTS-Diffusion produces synthetic time series on a pattern-by-pattern basis, iteratively employing the pattern generation module and the pattern evolution module, as outlined in Algorithm 2 in Appendix D.2. Given an initial segment sampled from the observed data, we start the generation of successive segments. At each position m , we predict the next pattern p_{m+1} , its length-scaling factor α_{m+1} , and magnitude-scaling factor β_{m+1} using the pattern evolution network. With these states, we generate the next segment \mathbf{x}_{m+1} by the pattern generation module. The time series then grows as more segments are generated and appended. This procedure is repeated until the synthetic data reaches the desired total length.

5 NUMERICAL EXPERIMENTS

We conduct numerical experiments to evaluate the performance of our FTS-Diffusion compared with alternatives, i.e., whether the generated data resemble real data and are useful for downstream tasks.

5.1 DATA AND EXPERIMENTAL SETTING

We run experiments on three different types of financial assets with varying characteristics: the Standard and Poor’s 500 index (S&P 500), the stock price of Google (GOOG), and the corn futures traded on the Chicago Board of Trade (ZC=F). S&P 500 and ZC=F span 1980-01-01 to 2020-01-01, and GOOG spans 2005-01-01 to 2020-01-01. We employ an 80/20 train-test split strategy, using the first 80% for training and the remaining 20% for testing. Importantly, neither FTS-Diffusion nor downstream models in our subsequential experiments have seen the test sets during the training phase, and all of our evaluation is on an out-of-sample basis. In finance, it is known that the raw asset prices follow a non-stationary random walk and are not well-behaved for statistical models. Instead, the returns, i.e., closing price changes in consecutive time intervals, remain relatively constant statistical properties (such as mean and variance) over time. Thus, we compare the return series generated by FTS-Diffusion to those by representative baselines: RCGAN (Esteban et al., 2017), TimeGAN (Yoon et al., 2019), and CSDI (Tashiro et al., 2021), whose details are in Appendix E.1

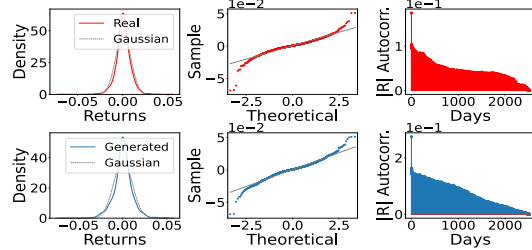


Figure 5: Real and generated time series of S&P 500 spanning 10 trading years, contain stylized facts of the heavy-tailed distribution (fat tails in density and QQ-plot, the first two columns, compared to the Gaussian) and decaying autocorrelations in absolute return (the last column).

⁴Note that we do not directly estimate the state transition probabilities of the Markov chain due to the large state space and data sparsity. Our NN-based approach has the advantage of generalizing from the training data to unseen states, making reasonable predictions even when faced with states not observed during training.

Table 1: Generated return distributions compared to observed data. The KS and AD statistics are floored/capped at 0/1 and 0.01/0.25, respectively. A higher value indicates better goodness of fit. Variation in the test statistic across multiple runs is shown with a +/- range.

Model	S&P500		GOOG		ZC=F	
	KS	AD	KS	AD	KS	AD
RCGAN	.189 \pm .006	.073 \pm .004	.185 \pm .006	.068 \pm .004	.179 \pm .006	.065 \pm .005
TimeGAN	.293 \pm .004	.115 \pm .006	.288 \pm .007	.108 \pm .005	.287 \pm .007	.103 \pm .005
CSDI (Generative)	.168 \pm .003	.069 \pm .002	.156 \pm .004	.067 \pm .003	.157 \pm .003	.065 \pm .003
TimeVAE	.219 \pm .006	.102 \pm .004	.201 \pm .005	.094 \pm .005	.215 \pm .005	.098 \pm .004
FTS-Diffusion	.327\pm.003	.128\pm.003	.324\pm.004	.119\pm.002	.325\pm.003	.121\pm.003

5.2 PROPERTIES OF THE SYNTHETIC TIME SERIES

The synthetic financial time series should inherit the stylized facts (Cont, 2001; Barberis & Shleifer, 2003) of asset returns, and resemble the distribution of observed data to a high degree of fidelity.

Stylized facts of financial time series. The empirical properties of financial time series have been studied extensively in the literature, which is often referred to as stylized facts (Cont, 2001; Barberis & Shleifer, 2003). The empirical studies reveal that asset returns have heavy tails, and the autocorrelation of absolute returns decays slowly over time. We assess whether the synthetic time series adhere to these stylized facts in Fig. 5. Indeed, the synthetic series exhibit significant heavy tails in their distribution and gradual decay in the autocorrelation of absolute returns, conforming to the aforementioned stylized facts. These results suggest that our approach is capable of generating synthetic financial time series that preserve the essential properties of observed data.

Distribution comparison. We also evaluate the discrepancy between the distribution of the synthetic time series and that of observed data, using the Kolmogorov–Smirnov (KS) test and the Anderson–Darling (AD) test as evaluation metrics. These tests estimate the goodness of fit between the synthesized distribution and the distribution of actual returns. For both tests, a larger test statistic indicates a higher degree of similarity between the distributions. The KS test is more sensitive to differences in the center of the distribution, whereas the AD test is more aware of the tails of the distribution. Table 1 demonstrates that our FTS-Diffusion learns a quantitatively closer distribution to the observed data, compared to other baselines. This result further confirms the efficacy of our approach in generating financial time series that resemble the observed data. More quantitative results using other metrics are included in Appendix E.2

5.3 DOWNSTREAM PREDICTION ANALYSIS OF THE SYNTHETIC TIME SERIES

We expand “Training on Synthetic, Test on Real” (Esteban et al., 2017) and design two new settings to evaluate the usefulness of the synthetic data for downstream tasks. Specifically, we focus on the task of prediction and implement an LSTM-based downstream predictive model. This structure is a prevalent choice in the literature (Yoon et al., 2019; Jeon et al., 2022; Remlinger et al., 2022). The downstream model is employed to predict the next data point in the series, using the 64 previous historical values as input (see Appendix E.3 for the additional five-day ahead prediction). We compute the mean absolute percentage error (MAPE) averaged over multiple runs.

Training on Mixture, Test on Real (TMTR). In this setting, we train the downstream predictive model on a dataset that combines observed and synthetic data in different proportions. For instance, a dataset with a mixing proportion of (30%, 70%) would be composed of 30% of data sampled from the observed data and 70% of data synthesized by the generative model. We test the predictive model on the test set sampled from the observed data which had not been seen by the generative model. If the synthetic data resemble the observed data, the predictive power of the downstream model trained on datasets with different mixing proportions should remain similar. Fig. 6(a) shows the results of the TMTR experiment for the one-day forecast on the three assets. The predictive accuracy is remarkably consistent across all mixing proportions, when synthetic data are generated using FTS-Diffusion. In comparison, the predictive accuracy deteriorates (large MAPEs) as the proportion of observed data decreases, when synthetic data are generated using RCGAN, TimeGAN, or CSDI.

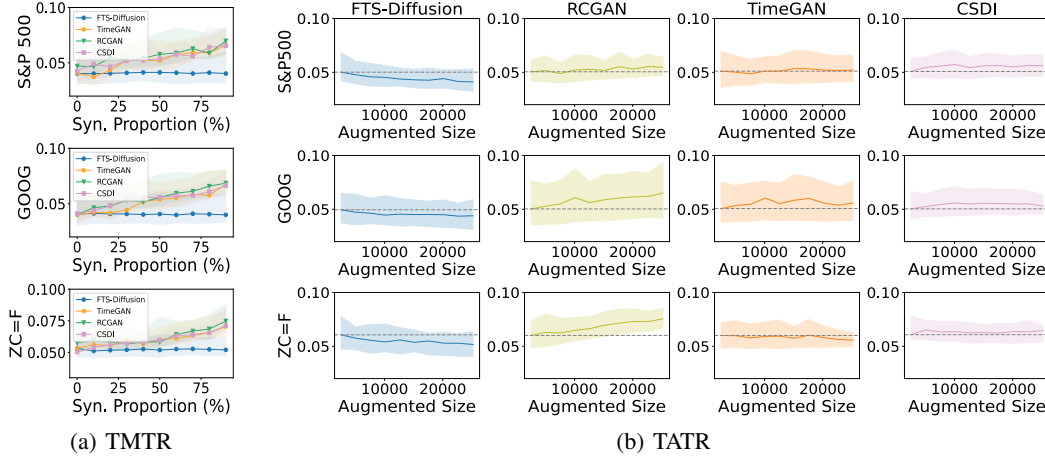


Figure 6: Prediction errors of the downstream model trained under the TMTR and TATR settings. Our FTS-diffusion maintains a comparable level of prediction accuracy across all mixing proportions of synthetic data and reduces the prediction errors by augmenting the observed dataset. Solid lines and shaded bands in each subfigure represent the average error and the 95% confidence interval over multiple runs, respectively. Dashed lines in each TATR test mark the initial prediction errors.

Thus, FTS-Diffusion is capable of generating synthetic time series sufficiently similar to actual data to uphold the performance of a downstream prediction task, whereas other models cannot.

Training on Augmentation, Test on Real (TATR). We initialize the training set with 10 years of observed data. We then iteratively append additional 10 years of synthetic data and evaluate the resulting performance of the downstream predictive model for a one-day ahead forecast. The results in Fig. 6(b) show a clear downward trend in the prediction error as more synthetic data from FTS-Diffusion is added to the training set. Appending 100 years of synthetic data reduces the MAPE by 17.9%, 15.3%, and 17.4% on the three assets, respectively. In contrast, the prediction error either increases or largely remains the same when synthetic data are generated by other baselines. These results indicate that FTS-Diffusion can effectively alleviate the problem of data shortage by augmenting the training set with sufficient synthetic samples. The ablation study is provided in Appendix E.4 to evaluate whether the model designs in FTS-Diffusion all serve a useful purpose.

6 CONCLUDING REMARK

We present *FTS-Diffusion*, a generative framework, for synthesizing financial time series with irregular and scale-invariant patterns. We break down the challenging financial time series generation into a *pattern recognition-generation-evolution* scheme. To facilitate this process, we design three dedicated modules: (i) a pattern recognition module leveraging our proposed SISC algorithm carefully designed to identify these patterns, (ii) a pattern generation module using a diffusion-based network to synthesize the segments of patterns, and (iii) a pattern evolution network to assemble generated segments with proper temporal evolution between consecutive patterns. Experimental results confirm the effectiveness of FTS-Diffusion in synthesizing financial time series that resemble observed data in distribution and their usefulness for downstream tasks. To the best of our knowledge, this is the first work in generating intricate yet crucial time series that encompass irregular and scale-invariant patterns, holding the potential for diverse applications across domains beyond finance.

This work offers a new and expansive perspective on complex (financial) time series generation by exploring their intrinsic properties, such as irregularity and scale-invariance. A promising direction for future research would be to extend our work to more challenging problem settings, e.g., the multivariate modeling that encompasses interactive dependencies across multiple time series. Our approach can handle potential distribution shifts arising from changes in the number of patterns, and an extension may be able to address shifts in transitions between patterns. One could also strengthen the theoretical guarantee of the generation quality. We leave these ideas for future research.

REFERENCES

- Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision—ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, pp. 622–637. Springer, 2019.
- Jushan Bai and Peng Wang. Conditional markov chain and its application in economic time series analysis. *Journal of applied econometrics*, 26(5):715–734, 2011.
- Nicholas Barberis and Andrei Shleifer. Style investing. *Journal of financial Economics*, 68(2): 161–199, 2003.
- Eoin Brophy. Synthesis of dependent multichannel ecg using generative adversarial networks. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pp. 3229–3232, 2020.
- Zhengping Che, Yu Cheng, Shuangfei Zhai, Zhaonan Sun, and Yan Liu. Boosting deep learning risk prediction with generative adversarial networks for electronic health records. In *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 787–792. IEEE, 2017.
- Yize Chen, Yishen Wang, Daniel Kirschen, and Baosen Zhang. Model-free renewable scenario generation using generative adversarial networks. *IEEE Transactions on Power Systems*, 33(3): 3265–3275, 2018.
- Rama Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative finance*, 1(2):223, 2001.
- Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. Timevae: A variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*, 2021.
- Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.
- Michael J Dueker. Markov switching in garch processes and mean-reverting stock-market volatility. *Journal of Business & Economic Statistics*, 15(1):26–34, 1997.
- Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.
- Benjamin Eysenbach, Alexander Khazatsky, Sergey Levine, and Russ R Salakhutdinov. Mismatched no more: Joint model-policy optimization for model-based rl. *Advances in Neural Information Processing Systems*, 35:23230–23243, 2022.
- Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.
- John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- Xin Huang and Duan Li. A two-level reinforcement learning algorithm for ambiguous mean-variance portfolio selection problem. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 4527–4533, 2021.
- Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/janner22a.html>.
- Jinsung Jeon, Jeonghak Kim, Haryong Song, Seunghyeon Cho, and Noseong Park. Gt-gan: General purpose time series synthesis with generative adversarial networks. *Advances in Neural Information Processing Systems*, 35:36999–37010, 2022.

- Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=a-xFK8Ymz5J>
- Yichong Leng, Zehua Chen, Junliang Guo, Haohe Liu, Jiawei Chen, Xu Tan, Danilo Mandic, Lei He, Xiangyang Li, Tao Qin, et al. Binauralgrad: A two-stage conditional diffusion probabilistic model for binaural audio synthesis. *Advances in Neural Information Processing Systems*, 35: 23689–23700, 2022.
- Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Using gans for sharing networked time series data: Challenges, initial promise, and open questions. In *Proceedings of the ACM Internet Measurement Conference*, pp. 464–483, 2020.
- Andrew W Lo, Harry Mamaysky, and Jiang Wang. Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The journal of finance*, 55(4): 1705–1765, 2000.
- Gerardo Manzo and Xiao Qiao. Deep learning credit risk modeling. *Forthcoming Journal of Fixed Income*, 2020.
- Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison W Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 2627–2633, 2017.
- Carl Remlinger, Joseph Mikael, and Romuald Elie. Conditional loss and deep euler scheme for time series generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8098–8105, 2022.
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.
- Poonam Somani, Shreyas Talele, and Suraj Sawant. Stock market prediction using hidden markov model. In *2014 IEEE 7th joint international information technology and artificial intelligence conference*, pp. 89–92. IEEE, 2014.
- Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csd: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.
- Robert L Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953.
- Mehmet Ozgur Turkoglu, William Thong, Luuk Spreeuwes, and Berkay Kicanaoglu. A layer-based sequential framework for scene generation with gans. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 8901–8908, 2019.
- Yuqing Wan, Xueyuan Gong, and Yain-Whar Si. Effect of segmentation on financial time series pattern matching. *Applied Soft Computing*, 38:346–359, 2016.
- Gefei Wang, Yuling Jiao, Qian Xu, Yang Wang, and Can Yang. Deep generative learning via schrödinger bridge. In *International Conference on Machine Learning*, pp. 10794–10804. PMLR, 2021.
- Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. Quant gans: Deep generation of financial time series. *Quantitative Finance*, 20(9):1419–1440, 2020.
- Hanwei Wu, Ather Gattami, and Markus Flierl. Conditional mutual information-based contrastive loss for financial time series forecasting. In *Proceedings of the First ACM International Conference on AI in Finance*, pp. 1–7, 2020.
- Yumo Xu and Shay B Cohen. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1970–1979, 2018.

- Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 5508–5518, 2019.
- Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.
- Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. Adversarially learned anomaly detection. In *2018 IEEE International conference on data mining (ICDM)*, pp. 727–736. IEEE, 2018.
- Chi Zhang, Sanmukh R Kuppannagari, Rajgopal Kannan, and Viktor K Prasanna. Generative adversarial network for synthetic time series data generation in smart grids. In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pp. 1–6. IEEE, 2018.
- Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.