Temporal-Spectral Shifted Diffusion Models for Time Series Forecasting

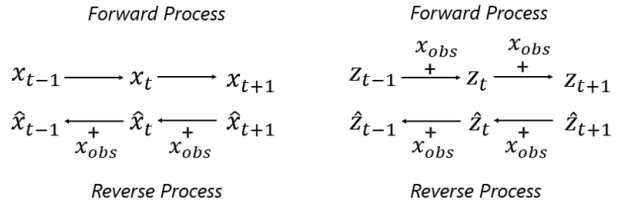
Anonymous Authors[1]

Abstract

The diffusion models have achieved significant success in conditional generation tasks, leading to recent research attention on leveraging diffusion models for time series forecasting. However, existing studies focused only on predicting time series in the time domain. To utilize the crucial characteristics in frequency domain for time series forecasting, we propose the time-frequency shifted diffusion model (TF-SHIFTER). Our approach accomplishes the prediction task by learning the distribution of time series in the time-frequency domain latent space. To better model the exact distribution of time series, we introduce additional shifts in the forward process of the diffusion model to optimize the diffusion trajectory. These shifts are related to the observed time series, implying that the prior distribution of the diffusion model incorporates the observed information. Experiments conducted on six real-world datasets validate the effectiveness of our proposed method.

1. Introduction

Time series forecasting plays a critical role in a variety of applications including weather forecasting (Karevan & Suykens, 2020; Hewage et al., 2020), finance forecasting (Dingli & Fournier, 2017; Cao et al., 2019), earthquake prediction (Lakshmi & Tiwari, 2009) and energy planning (Chou & Tran, 2018). Given that time series forecasting can be efficiently considered as a conditional generation task, numerous studies (Shen et al., 2018; Yoon et al., 2019) utilize generative models on time series forecasting tasks. Due to the remarkable progress diffusion models (Ho et al., 2020) have made in conditional generation tasks, more and more researchers concentrate on utilizing them in probabilistic time series forecasting. TimeGrad (Rasul et al., 2021) introduces noise to predict targets and progressively denoises through a transition kernel. With two conditioning mechanisms raised, TimeDiff (Shen & Kwok, 2023) can utilize diffusion models for non-autoregressive time series forecasting. CSDI (Tashiro et al., 2021) employs a specifically trained conditional score-based diffusion model for both probabilistic time series imputation and prediction.



(a) Forward and reverse processes in conventional time series diffusion models.

(b) Forward and reverse processes in our approach.

Figure 1. A simplified illustration of time series diffusion models with (a) conventional forward and reverse diffusion processes, (b) our temporal-spectral shifted diffusion models. x_{obs} denotes the observed time series, x_t and z_t denotes target time series and latent with frequency-domain information at time step t , respectively.

Present time series diffusion models are designed to learn the distribution of time series in the time domain exclusively. In these studies, the utilization of frequency-domain information was neglected. Therefore, existing time series diffusion models fail to capture frequency-domain related characteristics, such as seasonality, cyclic behavior, and periodic trends. However, such characteristics are crucial for time series forecasting tasks. Many studies (Owen et al., 2001; Yang et al., 2020; Zhou et al., 2022b) have demonstrated that capturing frequency-domain related characteristics can improve the accuracy of forecasting and the ability to forecast with longer horizons.

In this study, to better leverage frequency-domain related characteristics and achieve more accurate predictions, we propose a novel time-frequency (TF) domain diffusion model for time series forecasting tasks. Simply transforming time series into the time-frequency domain and forecasting the spectrogram with existing diffusion models will not yield improved results. Under the constraint of a limited-sized dataset, the diffusion model faces greater challenges in directly learning the exact distribution of time series in the time-frequency domain, which is always higher-dimensional

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

and more complex. To address this issue, as illustrated in Figure 1, we compare our contextualized diffusion models with conventional text-guided diffusion models. Two improvements are proposed in our method. Firstly, the spectrogram is encoded to latent space with Vector-Quantized Variational Autoencoder (VQ-VAE) (Van Den Oord et al., 2017). In other words, the time series is first transformed into the time-frequency domain and then encoded to a discrete latent space. The training difficulty of the diffusion model is reduced in low-dimensional latent space. Secondly, in our proposed diffusion model, the forward process gradually maps the target sequence to a prior distribution grounded in the observed time series instead of a Gaussian distribution. Specifically, we introduced an observation-related shift in the forward process during training and considered it as the starting point during sampling. We introduce this to optimize the starting point and trajectory of the diffusion process, facilitating learning more exact distribution. To more clearly illustrate the innovation of our method, we compare our approach with conventional time series diffusion models in Figure 1.

To summarize, our main contributions are summarized as follows:

- To the best of our knowledge, we are the first to incorporate a diffusion model in the time-frequency domain for time series forecasting, offering a novel perspective for future research.
- We optimize the diffusion trajectory by introducing additional shifts in the forward process. In our methodology, we model the distribution related to observations, rather than introducing the Gaussian distribution, as priors. This contributes to the improvement of prediction quality.
- We conducted experiments on six real-world datasets and provided a comprehensive presentation and analysis of the results using four metrics. The experimental results demonstrate that our approach achieves comparable or better results compared to baselines.

2. Related Work

2.1. Time Series Forecasting with Frequency Domain Information

In the research of signal processing and time series analysis, time-frequency domain analysis was first utilized as a feature extraction technique to enhance the performance of time series forecasting (Stankovic, 1994). Algorithms that combine frequency domain characteristics with traditional methods (*e.g.*, ARIMA(Box & Pierce, 1970)) can simultaneously leverage the advantages of moving average models in handling linear time series while reducing the generalization error in nonlinear time series (Yunus et al., 2015; Gupta & Kumar, 2020). Afterward, significant progress has been made by combining time-frequency domain transformations and deep neural networks (DNN). Convolutional neural networks(CNN) based methods utilize convolutional filters to capture long-term dependencies in the spectrogram (Shao et al., 2020; Zhang et al., 2023). Other methods (Yang et al., 2020; Wang et al., 2023) introduce frequency information into the Long Short-Term Memory(LSTM) frame to capture the frequency domain patterns, containing state components with different frequencies. Moreover, some studies that incorporate frequency domain information into Transformer have achieved success, such as(Zhou et al., 2022b; Chen et al., 2023). In recent years, approaches of combining Generative Adversarial Networks (GAN) with frequency domain information (Smith & Smith, 2020; 2021) have also achieved excellent results in the generation of time series. Despite all these successes, there is still no work introducing frequency domain information to the diffusion model for time series prediction tasks. In this paper, we are the first to implement the integration and conduct experiments with detailed analysis.

2.2. Diffusion Models for Time Series Forecasting

Recently, advancements have been made in the utilization of diffusion models for time series forecasting. In TimeGrad (Rasul et al., 2021) conditional diffusion model, with a denoising process guided by the hidden state, was first employed as an autoregressive approach for forecasting. CSDI (Tashiro et al., 2021) adopts a non-autoregressive generation strategy for faster sampling and incorporates self-supervised masking to guide the denoising process. SSSD (Alcaraz & Strodthoff, 2022) replaces the noise-matching network with a structured state space model and addresses the challenge in CSDI of dealing with time series with a large number of variables or lengths. TimeDiff (Shen & Kwok, 2023) incorporates future mix-up and autoregressive initialization for conditioning to a non-autoregressive framework. Nonetheless, all these time series diffusion models failed to leverage feature domain characteristics and started denoising from random noises, diverging from the approach in standard diffusion models. In this paper, our approach utilizes information in the frequency domain, and the starting point of the denoising process is a condition-related prior distribution rather than unrelated white noise. Our method has the capability to forecast time series with a significantly large number of variables or extended time series lengths because the denoising process happens in the latent space.

3. Preliminary

In this paper, We use $\boldsymbol{z}$ to refer to vectors in the latent space following (Rombach et al., 2022), and the letters "t" denote

the t-th step in the diffusion process.

Generative Time Series Forecasting. Suppose we have an observed time series $X = \{x_1, x_2, \cdots, x_n \,|\, x_i \in \mathbb{R}^d\}$, where n is the observed time length, d is the number of features per observation and x_i is the observation at time step i. The Y is the corresponding target time series $\{y_{n+1}, y_{n+2}, \cdots, y_{n+H} \,|\, y_j \in \mathbb{R}^{d'}\}$ ($d' \leq d$) , where H is the prediction horizon. The task of generative time series forecasting is to learn a density $p_\theta(Y|X)$ that best approximates $p(Y|X)$, which can be written as:

$$\min_{p_\theta} D\left(p_\theta(Y|X)||p(Y|X)\right), \tag{1}$$

where θ denotes parameters and D is some appropriate measure of distance between distributions. Given observation X the target time series can be obtained directly by sampling from $p_\theta(Y|X)$. Therefore, we obtain the time series $S = [X, Y]$.

Diffusion Models in Latent Space. The diffusion model progressively destructs latent $\boldsymbol{z}_0$ by injecting noise, then learns to reverse this process starting from $\boldsymbol{z}_T$ for sample generation. The forward process can be formulated as a Gaussian process with a Markovian structure:

$$\begin{aligned} q(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) &:= \mathcal{N}(\boldsymbol{z}_t; \sqrt{1-\beta_t}\boldsymbol{z}_{t-1}, \beta_t \boldsymbol{I}), \\ q(\boldsymbol{z}_t|\boldsymbol{z}_0) &:= \mathcal{N}(\boldsymbol{z}_t; \sqrt{\overline{\alpha}_t}\boldsymbol{z}_0, (1-\overline{\alpha}_t)\boldsymbol{I}), \end{aligned} \tag{2}$$

where $\beta_1, \ldots, \beta_T$ denotes fixed variance schedule with $\alpha_t := 1 - \beta_t$ and $\overline{\alpha}_t := \prod_{s=1}^{t} \alpha_s$. This forward process progressively injects noise to data until all structures are lost, which is well-approximated by $\mathcal{N}(0, \boldsymbol{I})$. The reverse diffusion process learns a model $p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ that approximates the true posterior:

$$p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t) := \mathcal{N}(\boldsymbol{z}_{t-1}; \mu_\theta(\boldsymbol{z}_t), \Sigma_\theta(\boldsymbol{z}_t)), \tag{3}$$

where μ_θ and Σ_θ are often computed by a UNet or a Transformer. Ho *et al.* (Ho et al., 2020) improve the diffusion training process and optimize following objective:

$$\mathcal{L}(\boldsymbol{z}_0) = \sum_{t=1}^{T} \mathop{\mathbb{E}}_{q(\boldsymbol{z}_t|\boldsymbol{z}_0)} ||\mu_\theta(\boldsymbol{z}_t, t) - \hat{\mu}(\boldsymbol{z}_t, \boldsymbol{z}_0)||^2, \tag{4}$$

where $\hat{\mu}(\boldsymbol{z}_t, \boldsymbol{z}_0)$ is the mean of the posterior $q(\boldsymbol{z}_{t-1}|\boldsymbol{z}_0, \boldsymbol{z}_t)$ which is a closed from Gaussian, and $\mu_\theta(\boldsymbol{z}_t, t)$ is the predicted mean of $p_\theta(\boldsymbol{z}_{t-1} \mid \boldsymbol{z}_t)$ computed by a neural network.

4. Method

In this section, we elucidate our proposed TF-SHIFTER in detail, which adapts the temporal-spectral shifted diffusion model to time series forecasting as in Figure 2. Our approach is divided into two components: representation and generation. In Section 4.1, we introduce how to represent time series in time-frequency latent space. To address this, we trained a VQ-VAE for the transformation. The Section 4.2 delves into the design of the shifted diffusion model, providing a comprehensive overview of both the forward and reverse processes. Finally, we theoretically derive the optimization objective and summarize the overall pipeline of the proposed framework (Section 4.3).

4.1. Representation of Time Series in Time-Frequency Latent Space

Transformation Process. To obtain the representation of time series in the time-frequency latent space, our approach involves two pivotal steps.

The first step involves transforming the time domain into the time-frequency domain, employing Short-Time Fourier Transform (STFT) and Inverse Short-Time Fourier Transform (ISTFT). Following the transformation, the data undergoes augmentation (Wen et al., 2020; Lee et al., 2023), increasing data dimensions. This augmentation process makes accurately learning the distribution of time series more challenging.

To address this issue, the time-frequency spectrum is encoded into a latent space. VQ-VAE is used as a basis in our work. VQ-VAE is known to produce sharper reconstructions (Van Den Oord et al., 2017) than Autoencoder (AE) (Bank et al., 2023) and Variational Autoencoder (VAE) (Kingma & Welling, 2013). The spectrogram is encoded into a continuous latent space and subsequently transformed into a discrete latent space through the codebook via the argmin process. In the argmin process, each continuous token is compared to every discrete token in the codebook in terms of the Euclidean distance and replaced with the closest discrete token, known as *quantization*. Given the relatively fixed patterns often observed in spectrograms (Sussillo et al., 2004), representing them in a learnable discrete latent space is considered reasonable. Subsequently, the decoders project the discrete latent spaces back into the time-frequency domains, equipped with corresponding zero-paddings. Finally, these domains are mapped back to the time domains via ISTFT.

Training Loss. The training framework for VQ-VAE can be found in the B.2. The optimization process involves enhancing the efficiency of an encoder, decoder, and codebook to proficiently compress the input into discrete latent. Simultaneously, the objective is to minimize information loss, which is evaluated by decoding the selected tokens and comparing the resulting output with the original input.

The codebook $\mathcal{C}$ consists of K discrete elements $\mathcal{C} = \{\boldsymbol{c}_k\}_{k=1}^{K}$, where each $\boldsymbol{c}_k \in \mathbb{R}^d$ and d denotes dimension size. The quantization process can be formulated as

$$\boldsymbol{z}_q = \mathop{\mathrm{argmin}}_{\boldsymbol{c}_k \in \mathcal{C}} \|\boldsymbol{z} - \boldsymbol{c}_k\|, \tag{5}$$

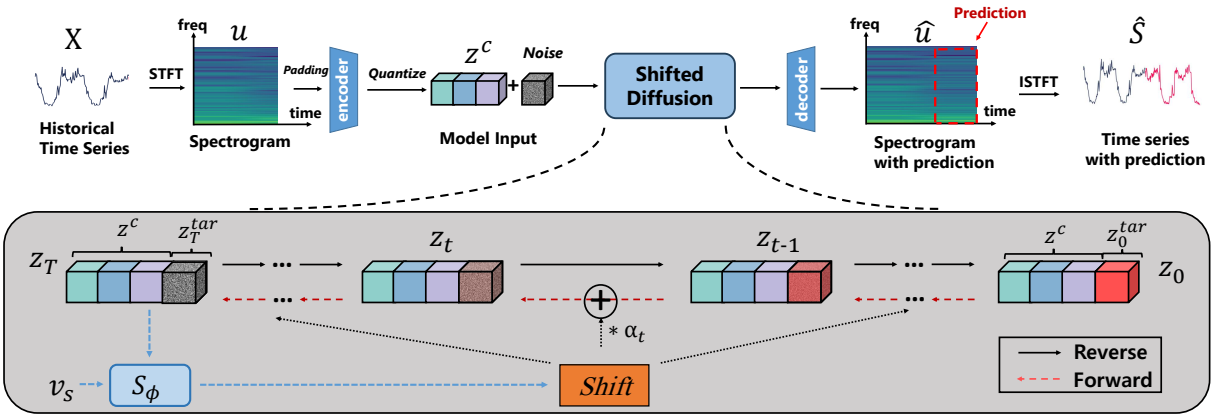


Figure 2. **Overview of TF-SHIFTER.** After acquiring the historical time series x, we proceed to input its time-frequency representation into the VQ-VAE encoder to derive the latent space representation $\boldsymbol{z}^c$. The shift is computed utilizing the side information v_s and $\boldsymbol{z}^c$. Within the graphical representation, the forward and backward processes of the diffusion model are depicted by red dashed and black solid lines, respectively. Notably, the shift exerts a direct influence on the initial stage of the entire backward process. Subsequently, we iterate through the backward process of the diffusion model until obtaining the predictive latent variable $\boldsymbol{z}_0^{tar}$. The latent variable $\boldsymbol{z}_0^{tar}$ is subsequently fed into the VQ-VAE decoder and transformed back into the time domain. All predicted targets, spanning from the latent space to the time-frequency domain and ultimately to the time domain, are distinctly marked in red. Further elucidation regarding v_s is provided in B.2.

where $\boldsymbol{z} \in \mathbb{R}^{d \times h \times w}$ denotes the activation map after the encoder, in which h and w represent the height and width of the map, respectively. The codebook-learning loss is then given by

$$\begin{aligned}\mathcal{L}_{\text{codebook}} = &\|\text{sg}[\boldsymbol{E}(\mathcal{P}(\boldsymbol{F}(S)))] - \boldsymbol{c}_k\|_2^2 \\ &+ \beta\|\boldsymbol{E}(\mathcal{P}(\boldsymbol{F}(S))) - \text{sg}[\boldsymbol{c}_k]\|_2^2,\end{aligned} \tag{6}$$

where S denotes the time series, $\boldsymbol{E}(\cdot)$ denotes VQ-VAE Encoder, $\boldsymbol{F}(\cdot)$ denotes STFT, $\text{sg}(\cdot)$ denotes the stop-gradient operation, $\mathcal{P}(\cdot)$ denotes the zero-padding operation preventing information leakage from the target window, and β is a weighting parameter for the commitment loss terms. Backpropagation through the non-differentiable quantization is achieved by copying the gradients from the decoder to the encoder.

The VQ loss also encompasses the reconstruction loss. In our approach, reconstruction tasks are performed in both the time and time-frequency domains, following a similar strategy as outlined in (Défossez et al., 2022). Thus, the reconstruction loss is given by

$$\mathcal{L}_{\text{recons}} = \|S - \hat{S}\|_2^2 + \|u - \hat{u}\|_2^2 \tag{7}$$

where $\hat{u}$ is the reconstruction of u, $\hat{S}$ are obtained by applying ISTFT to $\hat{u}$.

The total training objective for representation becomes:

$$\mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{codebook}} + \mathcal{L}_{\text{recons}}. \tag{8}$$

4.2. Time-Frequency Shifter for Time Series Diffusion

Motivation. Conventional time series diffusion models involve constructing a Markov chain in the forward process, gradually transforming observed data into a predefined prior distribution, typically the Gaussian distribution. While this approach excels in unconditional generation tasks, it may not be sustainable for time series forecasting tasks where observed time series is available. We believe that an appropriate diffusion model for time series forecasting should originate from a prior distribution grounded in the observed time series. In our proposed methodology, we integrate condition-related shift to incrementally modify the forward process. With a better prior distribution, the trajectory of the reverse process is optimized, making exact distribution learning more accessible.

Forward Process. We will provide a detailed explanation of the process involved in constructing such a Markov chain for time series forecasting. Let us consider the following forecasting problem: given a conditional observation X, we generate forecasting targets Y with side information v_s. v_s serves as additional features for the time series, and the computation process takes place within Appendix B.2. After training process in Section 4.1, Y and X are encoded into latent $\boldsymbol{z}^{tar}$ and $\boldsymbol{z}^c$. Then we define a shifting sequence $\{k_t\}_{t=1}^T$, which exhibits a monotonically increasing trend with the timestep t and fulfills the conditions $k_1 \to 0$ and $k_T \to 1$. Subsequently, the formulation of the transition distribution at timestep t is derived from this shifting sequence as outlined below (we highlight the critical parts of

our TF-SHIFTER in cyan):

$$q_\phi(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}^c, v_s) = \mathcal{N}(\boldsymbol{z}_t^{tar}; \boldsymbol{z}_{t-1}^{tar} + \alpha_t s_\phi(\boldsymbol{z}^c, v_s), \alpha_t \boldsymbol{I}) \tag{9}$$

where $[z^c, z_0^{tar}] = z_q$, $\alpha_t = k_t - k_{t-1}$ for $t > 1$ and $\boldsymbol{I}$ is the identity matrix. $s_\phi(\cdot)$ is the shifter network with trainable parameters ϕ, it takes the observed time series $\boldsymbol{z}^c$ and side information v_s as inputs and produces the shift with the same dimension as $\boldsymbol{z}_0^{tar}$. Then, we demonstrate that the marginal distribution at each timestep t is amenable to analytical integration, specifically,

$$q_\phi(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s) = \mathcal{N}(\boldsymbol{z}_t^{tar}; \boldsymbol{z}_0^{tar} + k_t s_\phi(\boldsymbol{z}^c, v_s), k_t \boldsymbol{I}) \tag{10}$$

To ensure a smooth transition between $\boldsymbol{z}_t^{tar}$ and $\boldsymbol{z}_{t-1}^{tar}$, to the standard deviation, denoted as $\bar{\alpha}_t$, is introduced. The rationale behind this lies in bounding the expected distance between $\boldsymbol{z}_t^{tar}$ and $\boldsymbol{z}_{t-1}^{tar}$ within $\bar{\alpha}_t$, given that the time series data falls within the range of $[0, 1]$ after pre-processing.

$$max[(\boldsymbol{z}_0^{tar} + k_t \hat{s}) - (\boldsymbol{z}_0^{tar} + k_{t-1}\hat{s})] = max[\alpha_t \hat{s}] < \alpha_t < \sqrt{\alpha_t}, \tag{11}$$

where $\hat{s} = s_\phi(\boldsymbol{z}^c, v_s)$, $max(\cdot)$ represents the point-wise maximizing operation. Besides the mean parameter, specifically $\boldsymbol{z}_0^{tar} + \alpha_t \hat{s}$, which contributes to the marginal distribution described in Equation (10). Moreover, the marginal distributions of $\boldsymbol{z}_1^{tar}$ and $\boldsymbol{z}_T^{tar}$ converge to $\delta_{\boldsymbol{z}_0^{tar}}(\cdot)$ and $\mathcal{N}(\cdot; \boldsymbol{z}^c, \boldsymbol{I})$, serving as approximations for the target and observation distributions, respectively. Through the deliberate construction of the Markov chain, it becomes feasible to address the forecasting task by reverse sampling from it given the observed time series $\boldsymbol{z}^c$.

Reverse Process. Following(Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) the objective of the reverse process is to infer the posterior distribution $p(\boldsymbol{z}^{tar}|\boldsymbol{z}^c)$ through the subsequent expression:

$$p(\boldsymbol{z}^{tar}|\boldsymbol{z}^c) = \int p(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c) \prod_{t=1}^{T} p^\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c) d\boldsymbol{z}_{1:T}^{tar}, \tag{12}$$

where $p(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c) \approx \mathcal{N}(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c, \boldsymbol{I})$, $p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)$ is the reverse transition kernel from $\boldsymbol{z}_t^{tar}$ to $\boldsymbol{z}_{t-1}^{tar}$ with a learnable parameter θ. Following most of the literature in the diffusion model, we adopt the assumption:

$$p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c) = \mathcal{N}(\boldsymbol{z}_{t-1}^{tar}; \mu_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t), \Sigma_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t)) \tag{13}$$

Combining Equation (9) and Equation (10), the targeted distribution $q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)$ in Equation (13) can be rendered tractable and expressed in an explicit form given below:

$$q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) = \mathcal{N}(\boldsymbol{z}_{t-1}^{tar}|\frac{k_{t-1}}{k_t}\boldsymbol{z}_t^{tar} + \frac{\alpha_t}{k_t}\boldsymbol{z}_0^{tar}, \frac{k_{t-1}}{k_t}\alpha_t \boldsymbol{I}) \tag{14}$$

The detailed calculation of this derivation is presented in Appendix A.1. Considering that the variance parameter is independent of $\boldsymbol{z}_t^{tar}$ and $\boldsymbol{z}^c$, we thus set $\Sigma_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t) = \frac{k_{t-1}}{k_t}\alpha_t \boldsymbol{I}$. Because $\boldsymbol{z}_t^{tar}$ is known, the mean parameter is parameterized as below:

$$\mu_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t) = \frac{k_{t-1}}{k_t}\boldsymbol{z}_t^{tar} + \frac{\alpha_t}{k_t} f_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t) \tag{15}$$

where f_θ is a deep neural network with parameter θ.

4.3. Training and Sampling

We introduce the derivation of the final optimization objective for the training TF-SHIFTER. Following the standard diffusion process, the objective function of TF-SHIFTER can be written as:

$$\begin{aligned}\mathcal{L}_{\theta,\phi} = \mathbb{E}_{q_\phi(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)} \Big[& D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) \| p(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c)) \\ & + \sum_{t>1} D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) \| p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)) \\ & - \log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c) \Big],\end{aligned} \tag{16}$$

The proof can be found in A.2. However, this training objective can be hard to stabilize (Nichol & Dhariwal, 2021). Thus we simplify the objective as follows:

$$\mathcal{L}_{\theta,\phi,t} = ||\boldsymbol{\mu}_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t) - \frac{\alpha_t}{k_t}\boldsymbol{z}_0^{tar} - \frac{k_{t-1}}{k_t}\boldsymbol{z}_t^{tar}||_2^2, \tag{17}$$

With Equation (15), the objective can be further simplified as:

$$\min_\theta \sum_t w_t ||f_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t) - \boldsymbol{z}_0^{tar}||_2^2 \tag{18}$$

where $w_t = \frac{\alpha_t}{2k_t k_{t-1}}$. In practical experimentation, we observe empirically that excluding the weight w_t leads to a noticeable enhancement in performance, consistent with the findings in (Ho et al., 2020).

Recall that the target time series are generated in time frequency latent space at each step t. We provide the pseudo-code of the sampling process at the inference stage in Algorithm 1.

5. Experiments

5.1. Experimental Setup

Datasets. Following previous work (Zhou et al., 2021; Wu et al., 2021; Fan et al., 2022; Shen & Kwok, 2023), experiments are performed on six real-world time series datasets (Table 5) : (1) *Traffic*[1], which records the hourly road occupancy rates generated by sensors located on the freeways in

[1] http://pems.dot.ca.gov

Table 1. Performance comparisons on six real-world datasets in terms of MSE and MAE. The best is in bold, while the second best is underlined.

Dataset	Exchange		Wind		Electricity		Weather		Traffic		ETTm1	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
TF-SHIFTER (ours)	**0.015**	**0.073**	**0.880**	**0.673**	0.161	0.256	**0.294**	**0.304**	0.543	0.376	**0.332**	**0.364**
TimeDiff	0.018	0.091	0.896	0.687	0.193	0.305	0.311	0.312	0.564	0.384	0.336	0.372
TimeGrad	0.079	0.193	1.209	0.793	0.736	0.630	0.392	0.381	1.745	0.849	0.874	0.605
CSDI	0.077	0.194	1.066	0.741	0.379	0.579	0.356	0.374	0.813	0.539	0.529	0.442
SSSD	0.061	0.127	1.188	0.778	0.255	0.363	0.349	0.350	0.642	0.398	0.464	0.406
D_3VAE	0.200	0.301	1.118	0.779	0.286	0.372	0.315	0.380	0.928	0.483	0.362	0.391
Fedformer	0.133	0.233	1.113	0.762	0.238	0.341	0.342	0.347	0.591	0.385	0.426	0.413
FreTS	0.039	0.140	1.004	0.703	0.269	0.371	0.351	0.354	0.903	0.482	0.397	0.412
FiLM	0.016	0.079	0.984	0.717	0.210	0.320	0.327	0.336	0.628	0.398	0.347	0.374
Autoformer	0.056	0.167	1.083	0.756	0.201	0.313	0.360	0.354	0.688	0.392	0.565	0.496
Pyraformer	0.032	0.112	1.061	0.735	0.273	0.379	0.394	0.385	0.659	0.390	0.493	0.435
Informer	0.073	0.192	1.168	0.772	0.292	0.383	0.385	0.364	0.664	0.391	0.673	0.542
Transformer	0.062	0.178	1.201	0.785	0.328	0.405	0.388	0.370	0.671	0.410	0.992	0.592
SCINet	0.038	0.137	1.055	0.732	0.171	0.280	0.329	0.344	0.434	0.335	0.359	0.389
DLinear	0.022	0.102	0.899	0.686	0.215	0.336	0.488	0.444	0.389	0.268	0.345	0.378
NLinear	0.019	0.091	0.989	0.706	**0.147**	**0.239**	0.313	0.328	0.430	0.293	0.349	0.375
Depts	0.020	0.100	1.082	0.751	0.319	0.401	0.761	0.394	1.019	0.568	0.380	0.412
NBeats	0.016	0.081	1.069	0.741	0.269	0.370	1.344	0.420	**0.373**	**0.265**	0.391	0.409

Algorithm 1 Sampling Algorithm of TF-SHIFTER

Input: Observed time series X, sampling steps t_1^T, shifting sequence k_t, the learned VA-VAE encoder$E$ and decoder D, the learned shifter model s_ϕ, the learned reverse model f_θ.
Output: The forecasting $\hat{Y}$.
Computing side information v_s,
Using VQ-VAE encoder $\boldsymbol{z}^c = E(X)$
Computing shift $\hat{s} = s_\phi(\boldsymbol{z}^c, v_s)$
Sample $\hat{z}_T^{ta} \sim \mathcal{N}(0, \boldsymbol{I})$
Reverse process start point $z_T^{ta} = \hat{z}_T^{ta} + \hat{s}$
for $t = T, T-1, \cdots, 1$ **do**
  $\alpha_t = k_t - k_{t-1}$
  Computing mean $\mu_t = \frac{k_{t-1}}{k_t}\boldsymbol{z}_t^{tar} + \frac{\alpha_t}{k_t} f_\theta(\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c, t)$
  Computing variance $\Sigma_t = \frac{k_{t-1}}{k_t}\alpha_t \boldsymbol{I}$
  Sampling $\hat{\boldsymbol{z}}_{t-1}^{tar} \sim \mathcal{N}(\boldsymbol{z}_{t-1}^{tar}; \mu_t, \sigma_t)$
end for
Decoding $\hat{Y} = \text{ISTFT}(D(\hat{\boldsymbol{z}}_0^{tar}))$

the San Francisco Bay area; (2) *Electricity*[2], which includes the hourly electricity consumption data from 321 clients over a span of two years.; (3) *Weather*[3], which documents 21 meteorological indicators at 10-minute intervals spanning from 2020 to 2021.; (4) *Exchange* (Lai et al., 2018), which describes the daily exchange rates of eight countries (Australia, British, Canada, Switzerland, China, Japan, New Zealand, and Singapore); (5) *ETTm1* (Zhou et al., 2021), which involves two years of electricity transformer temperature data collected in China at 15-minute intervals; (6) *Wind* (Li et al., 2022), which contains wind power records from 2020-2021. The details of datasets is shown in Appendix B.

Baseline Methods. We compare three kinds of time series forecasting methods. Our baselines include (1) Time series diffusion models, including TimeGrad (Rasul et al., 2021), CSDI (Tashiro et al., 2021), SSSD (Alcaraz & Strodthoff, 2022), D^3VAE (Li et al., 2022), TimeDiff (Shen & Kwok, 2023); (2) Recent time series forecasting methods with frequency information, including FiLM (Zhou et al., 2022a), Fedformer (Zhou et al., 2022b) and FreTS (Yi et al., 2023) ; (3) Time series transformers, including Autoformer (Wu et al., 2021), Pyraformer (Liu et al., 2021), Informer (Zhou et al., 2021) and the standard Transformer (Vaswani et al., 2017); (4) Other popular methods, including SciNet (Liu et al., 2022), Nlinear (Zeng et al., 2023) DLinear (Zeng et al., 2023), LSTMa (Bahdanau et al., 2015), Depts (Fan et al., 2022) and NBeats (Oreshkin et al., 2019).

Evaluation Metric. To comprehensively assess our proposed methodology, our experiment employs four metrics: (1) Probabilistic forecasting metrics, including Quantile Interval Coverage Error (QICE) (Han et al., 2022), Continuous Ranked Probability Score (CRPS) on each time

[2] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[3] https://www.bgc-jena.mpg.de/wetter/

Table 2. Part of performance comparisons on six real-world datasets in terms of QICE and CRPS. The best is in bold, while the second best is underlined. The full result is at Appendix C

Dataset	Exchange		Wind		Electricity		Weather		Traffic		ETTm1	
Metric	QICE	CRPS	QICE	CRPS	QICE	CRPS	QICE	CRPS	QICE	CRPS	QICE	CRPS
TF-SHIFTER (ours)	**4.332**	**0.339**	**6.980**	**0.920**	**5.291**	0.397	**3.814**	**0.324**	3.924	**0.299**	**3.742**	**0.374**
TimeGrad	4.479	0.590	7.567	0.923	5.396	0.491	7.302	0.411	3.784	0.369	5.374	0.605
CSDI	4.337	0.397	7.716	0.941	5.401	0.480	5.156	0.354	**3.429**	0.360	5.029	0.482
SSSD	5.121	0.427	7.601	0.981	5.711	0.429	9.409	0.350	3.792	0.401	4.864	0.556
D_3VAE	10.356	0.401	11.118	0.979	13.593	**0.389**	12.985	0.381	12.928	0.483	12.162	0.431

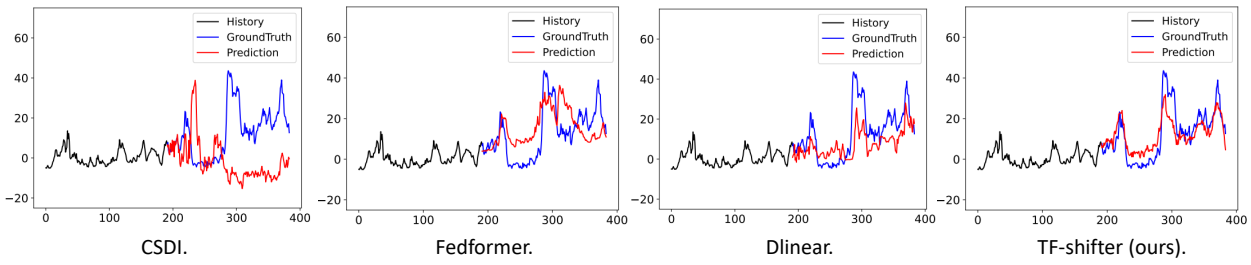


Figure 3. Visualizations on *wind* by CSDI, Fedformer, Dlinear and the proposed TF-SHIFTER.

series dimension (Matheson & Winkler, 1976). (2) Distance metrics, Mean Squared Error (MSE), and Mean Average Error(MAE) are employed to measure the distance between forecasting results and ground truth. Details of the metrics are in Appendix B.4.

Implementation Details. For training both VQ-VAE and the diffusion model, we utilized the Adam optimizer with an initial learning rate of 10^{-3}. During the training process of VQ-VAE, the batch size is set to 128, the weight decay is $1e^{-5}$, and the training spans 1000 epochs. In the training process of the shifted diffusion, the batch size is 64, and training includes early stopping for a maximum of 200 epochs. We adopt the transformer architecture from (Tashiro et al., 2021) as f_θ and the transformer architecture from [4] as the shift network s_ϕ. The number of diffusion steps T is set to 100, and the linear variance schedule starts from $K_0 = 0.001$ to $K_T = 0.999$. The history length is selected from (96, 192, 336, 720) using the validation set. All experiments are conducted on an Nvidia RTX A6000 GPU with 40GB memory. More model details can be found in Appendix B.2.

5.2. Main Results

Table 1 and Table 2 present the primary results of our experiments. It is obvious that our approach outperforms existing time series diffusion models. In comparison to other time series forecasting methods, our approach demonstrated superior performance on four out of six datasets, with competitive results on the remaining two datasets. Notably, CSDI encounters memory issues on the *Traffic* and *Electricity* datasets. Despite employing the same Transformer structure as CSDI, our approach conducts the diffusion process in the latent space, thereby alleviating the high training cost and the risk of memory overflow associated with directly forecasting long multivariate time series.

[4] https://github.com/lucidrains/linear-attention-transformer

Table 3. MSEs by different variants of the conditioning network.

In TF domain	With VQ-VAE	With shifter	*Weather*	*Exchange*	*Wind*
			0.356	0.077	1.066
✓			0.357	0.076	1.084
	✓		0.353	0.070	1.061
		✓	0.332	0.058	1.039
✓	✓		0.309	0.029	1.003
✓		✓	0.312	0.031	0.937
	✓	✓	0.323	0.044	0.997
✓	✓	✓	**0.294**	**0.015**	**0.880**

Figure 3 compares the forecasting results obtained by the four forecasting models, namely, CSDI, Fedformer, DLinear, and our proposal, on a randomly selected *wind* sample. While CSDI provides accurate predictions in the first few time points, its longer-term predictions deviate significantly from the ground truth. The other three methods capture both the trend and periodic patterns, while our approach provides higher-quality predictions than the others.

5.3. Model Analysis

Ablation Studies of Model Architecture. To evaluate the effectiveness of VQ-VAE, the shifter, and frequency information, we conduct a comprehensive evaluation by comparing the full version of TF-SHIFTER with seven variants on

three datasets. The testing Mean Squared Errors (MSEs) are presented as the results in Table 3. The symbol ✓indicates the utilization of the mechanism in the network, while a blank denotes the absence of the mechanism. As observed, directly forecasting the spectrogram with diffusion models results in flat or even deteriorated performance. However, incorporating VQ-VAE (or shifter) in the time-frequency domain significantly enhances the model's performance. This highlights the effective utilization of frequency domain information in time series forecasting tasks.

Effect of Diffusion Steps. To mitigate the impact of uncertainty while preserving informative temporal patterns, configuring the diffusion steps is crucial. Inadequately small diffusion steps may yield a process lacking meaningful outcomes, while excessively large steps could lead to an uncontrollable diffusion. In this study, we investigate the influence of the number of diffusion steps, denoted as T, on *ETTm1* and *Weather*. We vary T within the range of 50 to 500. As depicted in Figure 4, the optimal value for T is approximately 100 for TF-SHIFTER, and increasing T beyond this value does not significantly improve the results.

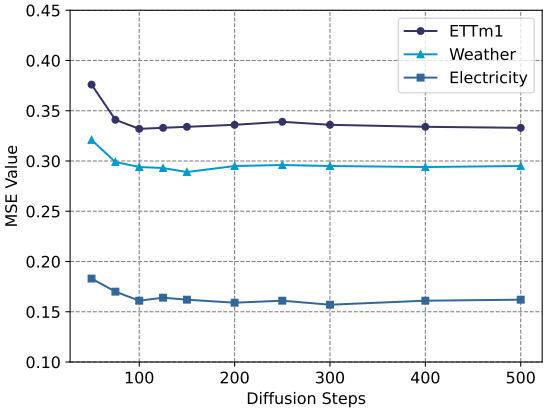


Figure 4. Comparisons of predictions with different T on three datasets.

Predicting z_0 vs Predicting ϵ. Following the formulation in Equation (18), our network is designed to forecast the latent variable z_0. Since some existing models (Rasul et al., 2021; Tashiro et al., 2021) have been trained by predicting an additional noise term ϵ, we conducted a comparative experiment to determine which approach is more suitable for our framework. Specifically, we maintained the network structure unchanged, only modifying the prediction target to be ϵ (including the shift). The results are presented in Table 4. Predicting z_0 proves to be more effective. This may stem from the challenge of simultaneously estimating shifts, generated noise, and the inherent non-linear noise present in real-world datasets.

Table 4. MSEs of two denoising strategies: Predicting z_0 vs predicting ϵ (including shift).

denoising strategy	*Weather*	*Electricity*	*Exchange*
z_0	**0.294**	**0.161**	**0.015**
ϵ	0.344	0.231	0.018

5.4. Inference Efficiency

In this experiment, we evaluate the inference efficiency of the proposed TimeDiff in comparison to other baseline time series diffusion models (TimeGrad, CSDI, SSSD). Figure 5 illustrates the inference time on the multivariate *weather* dataset with varying values of the prediction horizon (H). The results indicate that TimeDiff exhibits significantly faster inference times compared to the other models across all H values. Notably, TimeGrad is observed to be the slowest, attributed to its utilization of auto-regressive decoding.

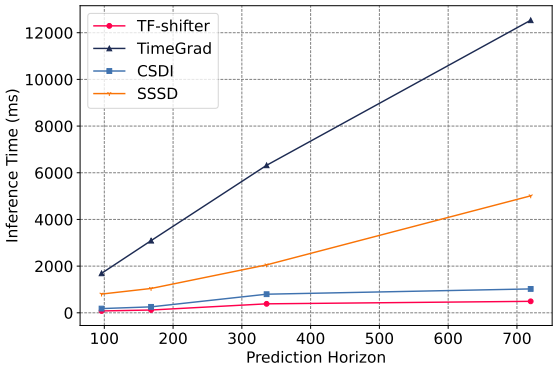


Figure 5. Inference time (ms) on the *Electricity* with different prediction horizon H.

6. Conclusion

In this paper, we propose TF-SHIFTER, an innovative framework that represents the first attempt to introduce frequency domain information into the diffusion model for time series prediction. By incorporating VQ-VAE and shifter mechanism, our model is better equipped to learn the accurate distribution of the spectrogram, enabling more precise predictions through the utilization of frequency domain information. Comprehensive experiments conducted on six real-world datasets demonstrate the outstanding performance of TF-SHIFTER in improving the quality of probabilistic predictions, underscoring its effectiveness.

7. Impact Statements

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Alcaraz, J. M. L. and Strodthoff, N. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*, 2022.

Bahdanau, D., Cho, K. H., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

Bank, D., Koenigstein, N., and Giryes, R. Autoencoders. *Machine learning for data science handbook: data mining and knowledge discovery handbook*, pp. 353–374, 2023.

Box, G. E. and Pierce, D. A. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American statistical Association*, 65(332):1509–1526, 1970.

Cao, J., Li, Z., and Li, J. Financial time series forecasting model based on ceemdan and lstm. *Physica A: Statistical mechanics and its applications*, 519:127–139, 2019.

Chen, Y., Liu, S., Yang, J., Jing, H., Zhao, W., and Yang, G. A joint time-frequency domain transformer for multivariate time series forecasting. *arXiv preprint arXiv:2305.14649*, 2023.

Chou, J.-S. and Tran, D.-S. Forecasting energy consumption time series using machine learning techniques based on usage patterns of residential householders. *Energy*, 165: 709–726, 2018.

Défossez, A., Copet, J., Synnaeve, G., and Adi, Y. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

Desai, A., Freeman, C., Wang, Z., and Beaver, I. Timevae: A variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*, 2021.

Dingli, A. and Fournier, K. S. Financial time series forecasting-a deep learning approach. *International Journal of Machine Learning and Computing*, 7(5):118–122, 2017.

Fan, W., Zheng, S., Yi, X., Cao, W., Fu, Y., Bian, J., and Liu, T.-Y. Depts: deep expansion learning for periodic time series forecasting. *arXiv preprint arXiv:2203.07681*, 2022.

Gupta, A. and Kumar, A. Mid term daily load forecasting using arima, wavelet-arima and machine learning. In *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*, pp. 1–5. IEEE, 2020.

Han, X., Zheng, H., and Zhou, M. Card: Classification and regression diffusion models. *Advances in Neural Information Processing Systems*, 35:18100–18115, 2022.

Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., and Liu, Y. Temporal convolutional neural (tcn) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing*, 24:16453–16482, 2020.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Karevan, Z. and Suykens, J. A. Transductive lstm for time-series prediction: An application to weather forecasting. *Neural Networks*, 125:1–9, 2020.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2020.

Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Lakshmi, S. S. and Tiwari, R. Model dissection from earthquake time series: A comparative analysis using modern non-linear forecasting and artificial neural network approaches. *Computers & Geosciences*, 35(2):191–204, 2009.

Lee, D., Malacarne, S., and Aune, E. Vector quantized time series generation with a bidirectional prior model. In *International Conference on Artificial Intelligence and Statistics*, pp. 7665–7693. PMLR, 2023.

Li, Y., Lu, X., Wang, Y., and Dou, D. Generative time series forecasting with diffusion, denoise, and disentanglement. *Advances in Neural Information Processing Systems*, 35: 23009–23022, 2022.

Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., and Xu, Q. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., and Dustdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*, 2021.

Matheson, J. E. and Winkler, R. L. Scoring rules for continuous probability distributions. *Management science*, 22 (10):1087–1096, 1976.

nadavbh12, Huang, Y.-H., Huang, K., and Fernández, P. nadavbh12/VQ-VAE. https://github.com/nadavbh12/VQ-VAE, 2021.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171. PMLR, 2021.

Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

Owen, J., Eccles, B., Choo, B., and Woodings, M. The application of auto–regressive time series modelling for the time–frequency analysis of civil engineering structures. *Engineering Structures*, 23(5):521–536, 2001.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Rasul, K., Seward, C., Schuster, I., and Vollgraf, R. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pp. 8857–8868. PMLR, 2021.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Shao, X., Pu, C., Zhang, Y., and Kim, C. S. Domain fusion cnn-lstm for short-term power consumption forecasting. *IEEE Access*, 8:188352–188362, 2020.

Shen, L. and Kwok, J. Non-autoregressive conditional diffusion models for time series prediction. *arXiv preprint arXiv:2306.05043*, 2023.

Shen, Z., Zhang, Y., Lu, J., Xu, J., and Xiao, G. Seriesnet: a generative time series forecasting model. In *2018 international joint conference on neural networks (IJCNN)*, pp. 1–8. IEEE, 2018.

Shen, Z., Zhang, M., Zhao, H., Yi, S., and Li, H. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 3531–3539, 2021.

Smith, K. E. and Smith, A. O. Conditional gan for timeseries generation. *arXiv preprint arXiv:2006.16477*, 2020.

Smith, K. E. and Smith, A. O. A spectral enabled gan for time series data generation. *arXiv preprint arXiv:2103.01904*, 2021.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Stankovic, L. A method for time-frequency analysis. *IEEE Transactions on Signal Processing*, 42(1):225–229, 1994.

Sussillo, D., Kundaje, A., and Anastassiou, D. Spectrogram analysis of genomes. *EURASIP Journal on Advances in Signal Processing*, 2004:1–14, 2004.

Tashiro, Y., Song, J., Song, Y., and Ermon, S. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, L., Mao, M., Xie, J., Liao, Z., Zhang, H., and Li, H. Accurate solar pv power prediction interval method based on frequency-domain decomposition and lstm model. *Energy*, 262:125592, 2023.

Wang, P., Olsen, K., and Bouaziz, W. lucidrains/vector-quantize-pytorch. https://github.com/lucidrains/vector-quantize-pytorch, 2022.

Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., and Xu, H. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*, 2020.

Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Yang, Z., Yan, W., Huang, X., and Mei, L. Adaptive temporal-frequency network for time-series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 34(4):1576–1587, 2020.

Yi, K., Zhang, Q., Fan, W., Wang, S., Wang, P., He, H., An, N., Lian, D., Cao, L., and Niu, Z. Frequency-domain mlps are more effective learners in time series forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Yoon, J., Jarrett, D., and Van der Schaar, M. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

Yunus, K., Thiringer, T., and Chen, P. Arima-based frequency-decomposed modeling of wind speed time series. *IEEE Transactions on Power Systems*, 31(4):2546–2556, 2015.

Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.

Zhang, Y. and Yan, J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023.

Zhang, Z., Chen, Y., Zhang, D., Qian, Y., and Wang, H. Ctfnet: Long-sequence time-series forecasting based on convolution and time–frequency analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Zhou, T., Ma, Z., Wen, Q., Sun, L., Yao, T., Yin, W., Jin, R., et al. Film: Frequency improved legendre memory model for long-term time series forecasting. *Advances in Neural Information Processing Systems*, 35:12677–12690, 2022a.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, pp. 27268–27286. PMLR, 2022b.

A. Theoretical Derivations

A.1. The Distributions in The Forward Process

First, we derive the explicit expressions for $q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ and $q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$, based on our shifted diffusion defined by Equation (9).

Lemma A.1. *For the forward process $q(\boldsymbol{z}_1^{tar}, \boldsymbol{z}_2^{tar}, ..., \boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s) = \prod_{t=1}^T q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$, if the transition kernel $q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ is defined as Equation (13), then the conditional distribution $q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ has the desired distribution as Equation (10), i.e., $\mathcal{N}(\boldsymbol{z}_t^{tar}; \boldsymbol{z}_{t-1}^{tar} + k_t s_\phi(\boldsymbol{z}^c, v_s), k_t \boldsymbol{I})$.*

Proof. We prove the lemma by induction. Suppose at time t, we have $q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ and $q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ admit the desired distributions as in Equations (9) and (10), respectively, then we need to prove that $q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s) = \mathcal{N}(\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar} + k_t s_\phi(\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, t), k_t \boldsymbol{I})$. We can re-write he conditional distributions of $\boldsymbol{z}_t^{tar}$ given $(\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ and $\boldsymbol{z}_{t-1}^{tar}$ given $(\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$ with the following equations:

$$\begin{aligned} \boldsymbol{z}_t^{tar} &= \boldsymbol{z}_{t-1}^{tar} + \alpha_t s_\phi(\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s) + \sqrt{\alpha_t}\epsilon_{t-1}, \\ &= \boldsymbol{z}_0^{tar} + \Sigma_{i=1}^t \alpha_t s_\phi(\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s) + \Sigma_{i=1}^t \sqrt{\alpha_t}\epsilon_i \end{aligned} \tag{19}$$

where ϵ_i are independent standard gaussian random variables. We can simplify Equation (19) as:

$$\boldsymbol{z}_t^{tar} = \boldsymbol{z}_0^{tar} + k_t s_\phi(\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s) + \sqrt{k_t}\epsilon_t \tag{20}$$

Then the marginal distribution of Equation (10) is obtained from Equation (20). □

Proposition A.2. *Suppose the distribution of forward process is defined by Equations (9) and (10), then at each time t, the posterior distribution $q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)$ is described by Equation (14)*

Proof. By the Bayes rule, $q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) = \frac{q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}$. By Equations (9) and (10), the numerator and denominator are both gaussian , then the posterior distribution is also gaussian and we can proceed to calculate its mean and variance:

$$\begin{aligned} q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) = &\frac{\mathcal{N}(\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar} + k_{t-1}\hat{s}, k_{t-1}\boldsymbol{I})}{\mathcal{N}(\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar} + k_t\hat{s}, k_t\boldsymbol{I})} \\ &* \mathcal{N}(\boldsymbol{z}_t^{tar}, \boldsymbol{z}_{t-1}^{tar} + \alpha_t\hat{s}, \alpha_t\boldsymbol{I}) \end{aligned}, \tag{21}$$

where $\hat{s}$ is an abbreviation form of $k_t s_\phi(\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c, v_s)$. Dropping the constants which are unrelated to $\boldsymbol{z}_0^{tar}, \boldsymbol{z}_t^{tar}, \boldsymbol{z}_{t-1}^{tar}$ and $\boldsymbol{z}^c$, we have:

$$\begin{aligned} q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) &\propto exp\left\{ -\frac{(\boldsymbol{z}_{t-1}^{tar} - \boldsymbol{z}_0^{tar} - k_{t-1}\hat{s})^2}{2k_{t-1}} + \frac{(\boldsymbol{z}_t^{tar} - \boldsymbol{z}_0^{tar} - k_t\hat{s})^2}{2k_t} \right. \\ &\left. - \frac{(\boldsymbol{z}_t^{tar} - \boldsymbol{z}_{t-1}^{tar} - \alpha_t\hat{s})^2}{2\alpha_t} \right\} \\ &= exp\left\{ C(\boldsymbol{z}_0^{tar}, \boldsymbol{z}_t^{tar}, \boldsymbol{z}^c) - \frac{1}{2}(\frac{1}{k_{t-1}} + \frac{1}{\alpha_t}) * (\boldsymbol{z}_{t-1}^{tar})^2 + \boldsymbol{z}_{t-1}^{tar} * \right. \\ &\left. [\frac{(\boldsymbol{z}_0^{tar} + k_{t-1}\hat{s})}{k_{t-1}} + \frac{(\boldsymbol{z}_t^{tar} - \alpha_t\hat{s})}{\alpha_t}] \right\} \\ &= exp\left\{ C(\boldsymbol{z}_0^{tar}, \boldsymbol{z}_t^{tar}, \boldsymbol{z}^c) - \frac{1}{2}(\frac{1}{k_{t-1}} + \frac{1}{\alpha_t}) * (\boldsymbol{z}_{t-1}^{tar})^2 + \boldsymbol{z}_{t-1}^{tar} * \right. \\ &\left. (\frac{1}{k_{t-1}}\boldsymbol{z}_0^{tar} + \frac{1}{\alpha_t}\boldsymbol{z}_t^{tar}]) \right\}, \end{aligned} \tag{22}$$

where $C(\boldsymbol{z}_0^{tar}, \boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)$ is a constant term with respect to $\boldsymbol{z}_{t-1}^{tar}$. Note that $(\frac{1}{k_{t-1}} + \frac{1}{\alpha_t}) = \frac{k_t}{k_{t-1}}$, and with some algebraic derivation, we can show that the gaussian distribution $q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)$ has:

$$\begin{aligned} variance: &\frac{k_{t-1}}{k_t}\alpha_t \boldsymbol{I} \\ mean: &\frac{\alpha_t}{k_t}\boldsymbol{z}_0^{tar} + \frac{k_{t-1}}{k_t}\boldsymbol{z}_t^{tar} \end{aligned} \tag{23}$$

□

This quadratic form induces the Gaussian distribution of Equation (14)

A.2. Upper Bound of The Likelihood

Here we show with our parameterization, that the objective function $\mathcal{L}_{\theta,\phi}$ Equation (18) is an upper bound of the negative log-likelihood of the data distribution.

Lemma A.3. *Based on the non-Markovian forward process* $q(\boldsymbol{z}_1^{tar}, \boldsymbol{z}_2^{tar}, ..., \boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c) = \prod_{t=1}^T q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)$ *and the conditional reverse process* $p_\theta(\boldsymbol{z}_0^{tar}, \boldsymbol{z}_1^{tar}, \boldsymbol{z}_2^{tar}, ..., \boldsymbol{z}_T^{tar}|\boldsymbol{z}^c) = p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c)\prod_{t=1}^T p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)$*, the objective function Equation* (16) *is an upper bound of the negative log likelihood.*

Proof.

$$\begin{aligned}
-\log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}^c) &\leq -\log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}^c) + \mathbb{E}_{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\left\{-\log \frac{p_\theta(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right\} \\
&= \mathbb{E}_{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\left\{-\log \frac{p_\theta(\boldsymbol{z}_{0:T}^{tar}|\boldsymbol{z}^c)}{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right\} \\
&= -\mathbb{E}_{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\left\{\log \frac{p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c)\prod_{t=1}^T p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)}{\prod_{t=1}^T q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right\} \\
&= -\mathbb{E}_{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\left\{\log p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c) + \sum_{t>1}\log \frac{p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_{t-1}^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)} + \log \frac{p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_1^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right\} \\
&= -\mathbb{E}_{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\left\{\log p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c) + \log \frac{p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_1^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right. \\
&\quad \left. + \sum_{t>1}\log \frac{p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)} * \frac{q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right\} \\
&= -\mathbb{E}_{q(\boldsymbol{z}_{1:T}^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\left\{\log \frac{p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c)}{q(\boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)} + \log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c) + \log \sum_{t>1}\frac{p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)}{q(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\right\} \\
&= D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)\|p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c)) - \mathbb{E}_{q(\boldsymbol{z}_1^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c) \\
&\quad + \sum_{t>1}\mathbb{E}_{q(\boldsymbol{z}_t^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)\|p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c))
\end{aligned} \tag{24}$$

□

Assume that the total diffusion step T is big enough and only a negligible amount of noise is added to the data at the first diffusion step, then the term $D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)\|p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c)) - \mathbb{E}_{q(\boldsymbol{z}_1^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)}\log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c)$ is approximately zero. Now with Lemma A.3, we have the following proposition:

Proposition A.4. *The objective function defined in Equation* (17) *is an upper bound of the negative log-likelihood.*

A.3. Achieving Better Likelihood with TF-shifter

Next, we show that the TF-SHIFTER is theoretically capable of achieving better likelihood compared to original DDPMs. As the exact likelihood is intractable, we aim to compare the optimal variational bounds for negative log-likelihoods. The objective function of TF-SHIFTER at time step t is $E_{q_\phi} D_{KL}(q_\phi(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c))$, and its optimal solution is

$$\begin{aligned}
&\min_{\phi,\theta} \mathbb{E}_{q_\phi} D_{KL}(q_\phi(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)) \\
&= min_\phi[min_\theta \mathbb{E}_{q_\phi} D_{KL}(q_\phi(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c))] \\
&\leq min_\theta \mathbb{E}_{q_{\phi=0}} D_{KL}(q_{\phi=0}(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)),
\end{aligned} \tag{25}$$

where $\phi = 0$ denotes setting the adapter network identical to 0, and thus $min_\theta \mathbb{E}_{q_{\phi=0}} D_{KL}(q_{\phi=0}(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c))$ is the optimal loss of origianl DDPMs objective at time t. Similar inequality can be obtained for t=1:

$$\begin{aligned}
&\min_{\phi,\theta} \mathbb{E}_{q_\phi} - \log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c) \\
&\leq \min_{\theta} \mathbb{E}_{q_{\phi=0}} - \log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c).
\end{aligned} \tag{26}$$

As a result, we have the following inequality by summing up the objectives at all time steps:

$$\begin{aligned}
&- \mathbb{E}_{q(\boldsymbol{z}_0^{tar})} \log p_\theta(\boldsymbol{z}_0^{tar}) \\
&\leq min_{\phi,\theta} \sum_{t>1} \mathbb{E}_{q_\phi} D_{KL}(q_\phi(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)) + \mathbb{E}_{q_\phi} - \log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c) + C \\
&\leq min_\theta \sum_{t>1} \mathbb{E}_{q_{\phi=0}} D_{KL}(q_{\phi=0}(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)||p_\theta(\boldsymbol{z}_{t-1}^{tar}|\boldsymbol{z}_t^{tar}, \boldsymbol{z}^c)) + \mathbb{E}_{q_{\phi=0}} - \log p_\theta(\boldsymbol{z}_0^{tar}|\boldsymbol{z}_1^{tar}, \boldsymbol{z}^c) + C
\end{aligned} \tag{27}$$

, where $C = \mathbb{E} D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}_T^{tar}|\boldsymbol{z}_0^{tar}, \boldsymbol{z}^c)\|p_\theta(\boldsymbol{z}_T^{tar}|\boldsymbol{z}^c))$ is a constant. Hence, the TF-SHIFTER has a tighter bound for the Negative Log-Likelihood (NLL), and is thus theoretically capable of achieving more accurate distribution, compared with the original DDPMs.

B. Implementation Details

B.1. Datasets Details

Due to varying sampling interval lengths across different datasets, employing the fixed set of prediction horizons 96, 192, 336, 720 for all datasets, as in (Zhou et al., 2021; Wu et al., 2021), may not be appropriate. For instance, the *Exchange* dataset consists of daily exchange rates. A prediction horizon of 720 corresponds to forecasting two years into the future, which may be excessive. Instead, we set the prediction horizon H to 14, corresponding to 2 weeks into the future, which is deemed more reasonable. Similarly, we set H = 168 for *ETTm1* (equivalent to 1 week into the future), H =672 for *Weather* (corresponding to 1 week), and so forth, as outlined in Table 5. It is noteworthy that some papers also choose the prediction length based on the dataset's sampling frequency. For example, Liu et al. (2021) and Zhang & Yan (2023) also use 168 (instead of 192) for *ETTm1* and *Traffic*.

In the diffusion process, we use the following 128-dimensions time-step embedding following previous works (Vaswani et al., 2017; Kong et al., 2020; Tashiro et al., 2021):

$$t_{embedding}(t) = \left(\sin(10^{0\cdot 4/63}t), \ldots, \sin(10^{63\cdot 4/63}t), \cos(10^{0\cdot 4/63}t), \ldots, \cos(10^{63\cdot 4/63}t)\right). \tag{28}$$

B.2. Network Architecture

STFT, ISTFT, and Convolutional Kernel and Stride Sizes. The STFT (Short-Time Fourier Transform) and ISTFT (Inverse Short-Time Fourier Transform) are implemented using [5] and [6], respectively. The window length of the short-time

[5] https://pytorch.org/docs/stable/generated/torch.stft.html
[6] https://pytorch.org/docs/stable/generated/torch.istft.html

Table 5. Summary of dataset statistics, including dimension, total observations, sampling frequency, and prediction length used in the experiments.

dataset	dim	#observations	freq.	H (steps)
Weather	21	52,696	10 mins	1 week (672)
ETTm1	7	69,680	15 mins	2 days (192)
Wind	7	48,673	15 mins	2 days (192)
Traffic	862	17,544	1 hour	1 week (168)
Electricity	321	26,304	1 hour	1 week (168)
Exchange	8	7,588	1 day	2 weeks (14)

Fourier transform is set to 8. Under this setting, the range of the frequency axis is [1, 2, 3, 4, 5]. Due to the default setting of the hop length as 2, the length of the time-frequency spectrogram becomes half of the given time series. The encoder consists of multiple downsampling layers, compressing the input into the latent space. The downsampling rate is determined by the convolutional kernel and stride size, typically using 2D Convolution layer(kernel size: 4, stride: 2, padding: 1) for a 2x downsampling at each downsampling layer. However, selecting these sizes is challenging due to variations in the time and frequency axis lengths across different datasets. Therefore, we let the downsampling layers only downsample along the time axis, rather than the frequency axis. Subsequently, the downsampling layers are transformed into 2D convolutional layers (kernel size: (3, 4), stride: (1, 2), padding: (1, 1)).

Our experiments found that smaller STFT windows, such as 4 or 8, achieve better performance. This is related to the amount of compression applied to the input data. Larger STFT windows result in a wider frequency axis and a shorter time axis, reducing the amount of compression as downsampling is applied only along the time axis. The global coherence of generated samples is often poorer at larger compression amounts (i.e., lower downsampling rates). We only reconstruct the prediction part and use all frequency bands to avoid reconstruction errors.

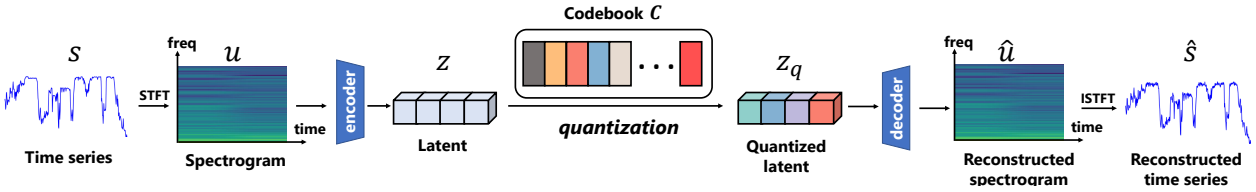


Figure 6. The framework of training VQ-VAE. We train VQ-VAE by minimizing the distance between reconstructed time series $\hat{X}$ and original time series X, as well as the distance between the reconstructed spectrogram $\hat{u}$ and the original spectrogram u.

VQ-VAE. We train the VQ-VAE using the framework depicted in 6. In our approach, we employ the same encoder and decoder architecture as outlined in the VQ-VAE paper, with their implementations referenced from (nadavbh12 et al., 2021). The encoder consists of 4 downsampling convolutional blocks (Conv2d – BatchNorm2d – LeakyReLU) followed by 4 residual blocks (LeakyReLU – Conv2d – BatchNorm2d – LeakyReLU – Conv2d). The downsampling convolutional layers and residual convolutional layers are implemented using two-dimensional convolutional layers (kernel size=(3,4), stride=(1,2), padding=(1,1)) and (kernel size=(3,3), stride=(1,1), padding=(1,1)), respectively. Prior to encoding, additional padding modules are introduced to prevent label leakage and shape difference between output and input.

The decoder similarly comprises 4 residual blocks followed by 4 upsampling convolutional blocks, mirroring the network details of the encoder. The codebook size K is set to 512, and the code dimension size aligns with the hidden dimension size of the encoder and decoder. For codebook learning loss, we adopt an alternative approach involving commitment loss and exponential moving average, as mentioned in (Wang et al., 2022). We set β to 1, and the exponential moving average decay rate to 0.8.

Transformers. Our approach employs the Transformer architecture from CSDI, with the distinction of expanding the channel dimension to 128. The network comprises temporal and feature layers, ensuring the comprehensiveness of the model in handling the time-frequency domain latent while maintaining a relatively simple structure. Regarding the transformer layer, we utilized a 1-layer Transformer encoder implemented in PyTorch (Paszke et al., 2019), comprising multi-head attention layers, fully connected layers, and layer normalization. We adopted the "linear attention transformer" package [7], to enhance computational efficiency. The inclusion of numerous features and long sequences prompted this decision. The

[7] https://github.com/lucidrains/linear-attention-transformer

package implements an efficient attention mechanism (Shen et al., 2021), and we exclusively utilized the global attention feature within the package.

Our shift network is even smaller, consisting of a transformer with only 4 heads and 32 dimensions. Its role is more about transmitting observed information to the shift while ensuring dimensional alignment between observation and targets.

Side Information. We utilize the combination of temporal embedding and feature embedding as side information v_s. We use 128-dimensions temporal embedding following previous studies (Vaswani et al., 2017):

$$s_{embedding}(s_l) = \left(\sin(s_l/\tau^{0/64}), \ldots, \sin(s_l/\tau^{63/64}), \cos(s_l/\tau^{0/64}), \ldots, \cos(s_l/\tau^{63/64})\right) \tag{29}$$

where $\tau = 10000$. Following (Tashiro et al., 2021), s_l represents the timestamp corresponding to the l-th point in the time series. This setup is designed to capture the irregular sampling in the dataset and convey it to the model. Additionally, we utilize learnable embedding to handle feature dimensions. Specifically, feature embedding is represented as 16-dimensional learnable vectors that capture relationships between dimensions. According to (Kong et al., 2020), we combine time embedding and feature embedding, collectively referred to as side information v_s.

The shape of v_s is not fixed and varies with datasets. Taking the Exchange dataset as an example, the shape of forecasting target Y is [Batchsize (64), 7(number of variables), 168 (time-dimension), 12 (time-dimension)] and the corresponding shape of v_s is [Batchsize (64), total channel(144(time:128 + feature:16)), 320 (frequency-dimension*latent channel), 12 (time-dimension)], and the shape of shift is [Batchsize (64), 64(latent channels), 5 (frequency-dimension), 12 (time-dimension)].

B.3. Baselines

Codes for the baselines are downloaded from the following. (i) TimeGrad: https://github.com/ForestsKing/TimeGrad; (ii) CSDI: https://github.com/ermongroup/CSDI; (iii) SSSD: https://github.com/AI4HealthUOL/SSSD; (iv) D^3VAE: https://github.com/ramber1836/d3vae; (v) FiLM: https://github.com/DAMO-DI-ML/NeurIPS2022-FiLM; (vi) Depts: https://github.com/weifantt/DEPTS; (vii) NBeats: https://github.com/ServiceNow/N-BEATS; (viii) SCINet: https://github.com/cure-lab/SCINet; (ix) Fedformer: https://github.com/DAMO-DI-ML/ICML2022-FEDformer; (x) Autoformer: https://github.com/thuml/Autoformer; (xi) Pyraformer: https://github.com/ant-research/Pyraformer; (xii) Informer: https://github.com/zhouhaoyi/Informer2020; (xiii) Transformer: https://github.com/thuml/Autoformer/blob/main/models/Transformer.py; (xiv) DLinear: https://github.com/ioannislivieris/DLinear; (xv) LSTM-a: https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html. (xvi) FreTS: https://github.com/aikunyi/FreTS.

B.4. Metrics

We will introduce the metrics in our experiments. We summarize them as below:

CRPS. CRPS (Matheson & Winkler, 1976) is a univariate strictly proper scoring rule which ´ measures the compatibility of a cumulative distribution function F with an observation x as:

$$CRPS(F, x) = \int_R (F(y) - \mathbb{1}_{(x \le y)})^2 dy \tag{30}$$

where $\mathbb{1}_{(x \le y)}$ is the indicator function, which is 1 if $x \le y$ and 0 otherwise. The CRPS attains the minimum value when the predictive distribution F same as the data distribution.

QICE. To enhance the assessment of uncertainty estimation capabilities in probabilistic multivariate time series forecasting tasks, we introduce Quantile Interval Coverage Error (QICE) (Han et al., 2022). With a sufficient number of $y_{0:M}$ samples, the first step involves dividing them into M quantile intervals (QIs), each with approximately equal sizes. Subsequently, quantile values corresponding to each QI boundary are determined:

$$QICE = \frac{1}{M}\sum_{m=1}^{M} |r_m - \frac{1}{M}|, \text{where } r_m = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}_{y_n \ge \hat{y_n}_n^{low_m}} \cdot \mathbb{1}_{y_n \le \hat{y_n}_n^{high_m}} \tag{31}$$

where $\hat{y_n}_n^{low_m}$ and $\hat{y_n}_n^{low_m}$ represent the low and high percentiles, respectively, of our choice for the predicted $y_{0:M}$ outputs given the input. In cases where the learned distribution accurately represents the true distribution, this measurement should closely align with the difference between the selected low and high percentiles.

MAE and MSE. MAE and MSE are calculated in the formula below, $\hat{Y}$ represents the predicted time series, and Y represents the ground truth time series. MAE calculates the average absolute difference between predictions and true values, while MSE calculates the average squared difference between predictions and true values. A smaller MAE or MSE implies better predictions.

$$\begin{aligned} MAE &= mean(|\hat{Y} - Y|) \\ MSE &= \sqrt{mean(|\hat{Y} - Y|)} \end{aligned} \tag{32}$$

C. More Experiment Results

Table 6 shows the complete results of Table 2.

Table 6. The complete results of QICE/CRPS on six real world datasets.

Dataset	Exchange		Wind		Electricity		Weather		Traffic		ETTm1	
Method	QICE	CRPS	QICE	CRPS	QICE	CRPS	QICE	CRPS	QICE	CRPS	QICE	CRPS
TF-SHIFTER (ours)	**4.332**	**0.339**	**6.980**	**0.920**	**5.291**	0.397	**3.814**	**0.324**	3.924	0.299	**3.742**	**0.374**
TimeGrad	4.479	0.590	7.567	0.923	5.396	0.491	7.302	0.411	3.784	0.369	5.374	0.605
CSDI	4.337	0.397	7.716	0.941	5.401	0.480	5.156	0.354	3.429	0.360	5.029	0.482
SSSD	5.121	0.427	7.601	0.981	5.711	0.429	9.409	0.350	3.792	0.401	4.864	0.556
D_3VAE	10.356	0.401	11.118	0.979	13.593	**0.389**	12.985	0.381	12.928	0.483	12.162	0.431
Fedformer	6.623	0.631	11.093	1.235	7.468	0.561	6.262	0.347	3.821	0.505	4.315	0.503
FreTS	7.839	0.440	10.004	0.943	8.139	0.634	6.471	0.354	8.512	0.602	4.437	0.522
FiLM	9.014	0.349	7.569	6.998	9.969	0.671	5.849	0.336	4.270	0.478	3.971	0.474
Autoformer	10.256	0.769	9.896	1.026	6.513	0.602	8.452	0.354	4.368	0.463	5.312	0.566
Pyraformer	11.432	0.532	9.532	0.994	7.432	0.732	10.369	0.385	4.119	0.460	6.311	0.505
Informer	10.173	0.631	12.031	1.065	12.351	0.749	9.411	0.364	4.204	0.591	8.936	0.631
Transformer	8.162	0.629	12.221	1.026	13.627	0.801	10.317	0.370	5.011	0.630	12.191	0.742
SCINet	10.831	0.624	11.125	0.997	5.642	0.499	6.262	0.344	4.011	0.505	6.139	0.531
DLinear	5.017	0.538	7.613	0.957	5.998	0.527	9.311	0.444	3.919	0.318	4.915	0.498
NLinear	5.931	0.481	9.019	0.974	5.510	0.419	6.131	0.328	5.161	0.373	4.62	0.495
Depts	5.058	0.520	9.113	1.001	8.357	0.803	11.132	0.394	12.519	0.712	4.917	0.527
NBeats	4.516	0.399	11.131	0.981	6.931	0.697	12.993	0.420	**3.420**	**0.291**	5.031	0.519

To emphasize our distribution estimation capabilities, we present the predicted median and visualize the 50% and 90% distribution intervals in Figure 7. We compare TF-SHIFTER with CSDI and TimeGrad. Clearly, our method has achieved the best results, while the other two methods performed poorly on the forecasting task with a large horizon.

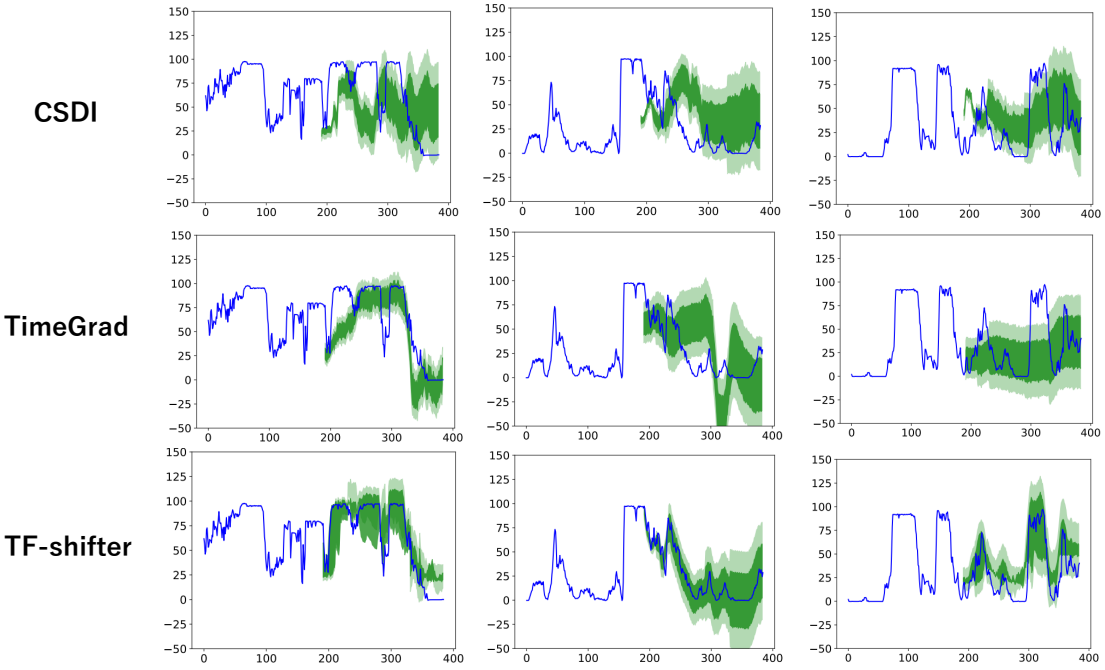


Figure 7. Comparison of prediction intervals for the Wind dataset. We display the predicted median and visualize the 50% and 90% distribution intervals, the blue line representing the test set ground truth.

8 represents the comparison between our predictions on the strongly periodic dataset electricity and those of CSDI. It can be observed that our predictions are significantly better compared to CSDI. Moreover, our method demonstrates a clear ability to accurately forecast the periodicity of time series.

D. More Model Analysis

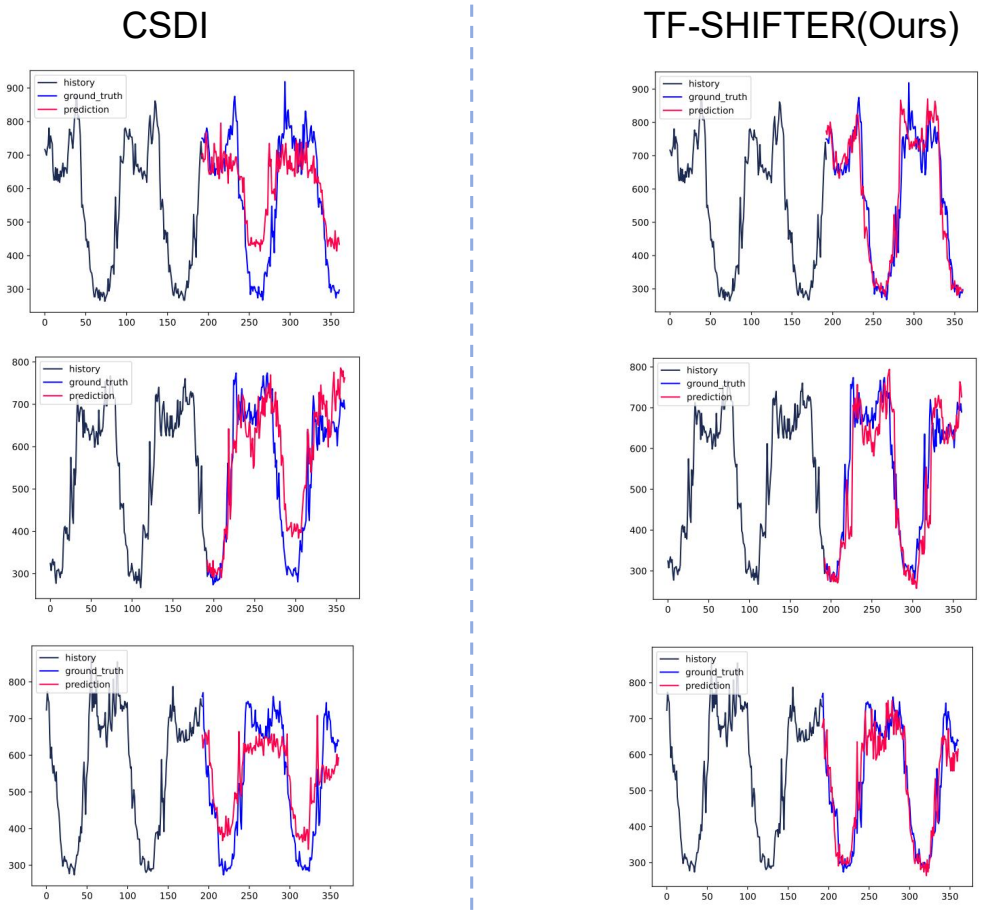


Figure 8. The comparison between TF-SHIFTER and CSDI on the electricity dataset, with our method shown on the right. In the figure, red lines represent the predicted results, blue lines represent the true values, and black lines represent the historical time series.

Noise Schedule. Our approach employs a shifting sequence k_t to determine the noise schedule in the diffusion process. Considering that the noise schedule may impact the performance of the diffusion model, we designed comparative experiments to select the noise schedule parameters. The subsequent exposition mainly revolves around the construction of the shifting sequence k_t. For the intermediate timesteps, i.e., $t \in [2, T-1]$, we propose a non-uniform geometric schedule for $\sqrt{k_t}$ as follows:

$$\sqrt{k_t} = \sqrt{k_1} \times b^{\beta_t}, t = 2, \ldots, T-1, \tag{33}$$

where

$$\beta_t = (\frac{t-1}{T-1})^p \times (T-1), b = exp[\frac{1}{2(T-1)} log \frac{k_T}{k_1}] \tag{34}$$

In order to determine the most suitable value for the parameter p in our model, we observed the results of the comparative experiments and ultimately chose 0.3 for application in the experiments.

Representation of Time Series. For the transformation from spectrum to LATENT, we conducted experiments using VQ-VAE, which was still a result obtained through comparison. We selected 1000 reconstructed sequences and their original sequences from the test set for analysis. We show t-SNE visualizations in Figure 9. VAE and TimeVAE(Desai et al., 2021) are two representative VAE baselines, respectively. In the figure, the synthetic samples generated from VQ-VAE are the best.

Table 7. Performance comparison of TF-SHIFTER on the *Exchange* under different p

Metric	MSE	MAE	QICE	CRPS
0.3	**0.015**	**0.073**	**4.332**	**0.339**
0.5	0.016	0.075	4.338	0.342
0.7	0.018	0.077	4.348	0.345
1.0	0.021	0.081	4.359	0.349
2.0	0.025	0.086	4.381	0.357

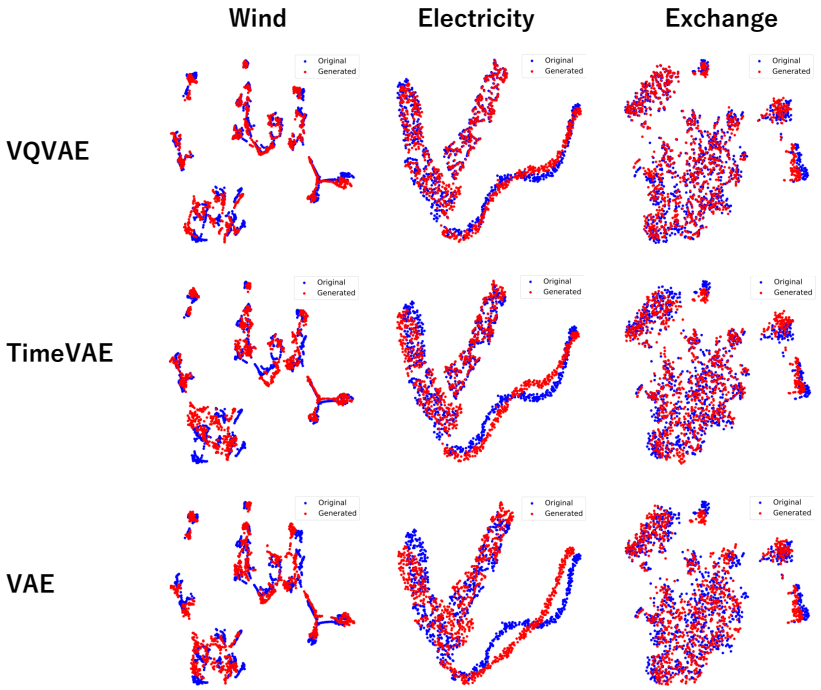


Figure 9. t-SNE plots for VQ-VAE (1st row), TimeVAE (2nd row), and VAE (3th row). Blue and red dots mean original and synthesized samples, respectively.