RDIT: Residual-based Diffusion Implicit Models for Probabilistic Time Series Forecasting

Chih-Yu Lai* Dept. of EECS, MIT Cambridge, MA, 02139 chihyul@mit.edu

Yu-Chien Ning Harvard University Cambridge, MA, 02138 bycning@hsph.harvard.edu Duane S. Boning
Dept. of EECS, MIT
Cambridge, MA, 02139
boning@mtl.mit.edu

Abstract

Probabilistic Time Series Forecasting (PTSF) plays a critical role in domains requiring accurate and uncertainty-aware predictions for decision-making. However, existing methods offer suboptimal distribution modeling and suffer from a mismatch between training and evaluation metrics. Surprisingly, we found that augmenting a strong point estimator with a zero-mean Gaussian, whose standard deviation matches its training error, can yield state-of-the-art performance in PTSF. In this work, we propose RDIT, a plug-and-play framework that combines point estimation and residual-based conditional diffusion with a bidirectional Mamba network. We theoretically prove that the Continuous Ranked Probability Score (CRPS) can be minimized by adjusting to an optimal standard deviation and then derive algorithms to achieve distribution matching. Evaluations on eight multivariate datasets across varied forecasting horizons demonstrate that RDIT achieves lower CRPS, rapid inference, and improved coverage compared to strong baselines. (Code can be downloaded at https://anonymous.4open.science/r/RDIT-16BB/)

1 Introduction

Time Series Forecasting (TSF) aims to predict future values of variables based on their historical observations. This challenging but important task can be widely applied to a variety of fields such as finance [1], healthcare [2], manufacturing [3], environmental science [4], and many others [5, 6, 7]. An emerging amount of literature focuses on implementing deep learning-based techniques for TSF [8, 9, 10]. PTSF can be applied to additionally model the distribution of the predictions for estimating the uncertainty [11], where several advantages include increased robustness, the ability of risk management, and the likelihood of anomaly detection [12, 13]. Diffusion-based generative models are a common approach for realizing PTSF [14, 15]. This involves iteratively denoising the data and arriving at some predictive pattern, where the end points of multiple trajectories form a distribution of the predictions. These models exhibit strong distribution modeling characteristics and have achieved state-of-the-art performance [16].

However, certain techniques shared among recent TSF methods inherently pose challenges for distributional modeling in PTSF [17, 18, 19]. Specifically, linear mappings may fail to capture nonlinear relationships or interactions that are critical for accurately modeling the distribution of the forecast [18], and channel independence can prevent the model from learning dependencies among variables, resulting in unrealistic uncertainty estimates [20]. While recent TSF methods do not strictly enforce the use of these mechanisms [21, 22, 23], integrating both point estimation and noise estimation within a single model often leads to a trade-off between prediction accuracy and diversity, making it difficult to model uncertainty effectively [24]. To address this issue, TMDM [24] separates the tasks of point-based (conditional mean) estimation and noise estimation: the former is predicted using a plug-and-play transformer-based model, and the latter is modeled using diffusion. This

separation allows any TSF model to be integrated for point estimation while enabling more targeted design of the noise estimation process. Recently, D3U [25] improves upon TMDM by incorporating a better-performing point estimator, residual modeling, and explicitly decoupling deterministic and uncertain components. It also adopts the DPM-Solver [26] for faster inference.

Another challenge is that as the CRPS is a widely used metric for evaluating PTSF, using the Mean Absolute Error (MAE) or Mean Squared Error (MSE) as the training loss can lead to a mismatch between the training objective and the evaluation metric. This misalignment may result in under-dispersed or overconfident predictions [27], suggesting that post-hoc adjustments to the noise estimates are necessary to optimize the final predictive distribution. Additionally, since the residuals contain inherent noise, it is easy for models to overfit to them, leading to a potential mismatch between the Predictive Interval Coverage Probability (PICP) of the predicted and true distributions. These challenges can result in suboptimal performance. What's more, we find that a strong baseline can be constructed by equipping a point estimator with a zero-mean Gaussian distribution fitted to the training residuals, i.e., $\mathcal{N}(0, \sigma_{trn}^2)$, which achieves excellent results in several settings (Section 4.3). This introduces a *race* between improving point estimation and uncertainty modeling. A robust PTSF algorithm should outperform its corresponding point estimator in both point-based and probabilistic metrics even when the point estimator is enhanced with a fitted Gaussian.

In this work, we propose **R**esidual-based **D**iffusion Implicit Models for Probabilistic Time Series Forecasting (RDIT), a framework that leverages point estimators for accurate point forecasting and noise estimation networks to model the residuals between the ground truth and point predictions. To address the distribution mismatch problem, we introduce two novel algorithms: 1) Error-aware Expansion (EAE), which mitigates the discrepancy between MAE and CRPS by optimizing the predicted variance, and 2) Coverage Optimization (CO), which calibrates on a hold-out validation set to adjust the final distribution. RDIT incorporates Denoising Diffusion Implicit Models (DDIMs) [28] for fast inference and introduces the PICP distance metric, which measures the total deviation from the target coverage across multiple quantiles (cf. Appendix C). Our experiments demonstrate that RDIT outperforms ten baselines across eight datasets and multiple prediction lengths.

Key contributions of this paper are:

- 1. We decouple point estimation and residual modeling into two stages—using TSF models to generate point estimates and applying diffusion to normalized residuals—to improve learnability by minimizing the discrepancy between the start and end of the diffusion process.
- 2. We propose two distribution-matching algorithms: (a) EAE that minimizes the CRPS under a fixed-mean Gaussian predictive distribution by optimizing the variance, and (b) CO, a coverage optimization algorithm that corrects mismatches between true and predictive coverage. Both contribute to improved performance.
- 3. We design a noise estimation network using bidirectional Mamba layers to effectively model temporal dependencies in the residuals, leveraging correlations between the denoising target and the input to enhance inductive bias.

2 Preliminaries

2.1 Denoising Diffusion Probabilistic Models (DDPMs)

DDPMs define a forward diffusion process where an initial data point \mathbf{r}^0 is gradually corrupted over K diffusion steps by adding Gaussian noise. At each step $k \in \{1, 2, \dots, K\}$, the data is transformed according to $q(\mathbf{r}^k|\mathbf{r}^{k-1}) = \mathcal{N}\left(\sqrt{1-\beta_k}\mathbf{r}^{k-1}, \beta_k\mathbf{I}\right)$, where $\beta_k \in [0,1]$ is a variance schedule parameter controlling the amount of noise added step k. By leveraging the properties of the Gaussian distribution and the variational approximation, the marginal distribution of \mathbf{r}^k given \mathbf{r}^0 is

$$q(\mathbf{r}^k|\mathbf{r}^0) = \mathcal{N}(\sqrt{\bar{\alpha}_k}\mathbf{r}^0, (1-\bar{\alpha}_k)\mathbf{I}), \text{ with } \bar{\alpha}_k = \prod_{s=1}^k \alpha_s, \ \alpha_s = 1-\beta_s.$$
 (1)

Thus, \mathbf{r}^k can be directly sampled from \mathbf{r}^0 using:

$$\mathbf{r}^{k} = \sqrt{\bar{\alpha}_{k}} \mathbf{r}^{0} + \sqrt{1 - \bar{\alpha}_{k}} \boldsymbol{\varepsilon}^{k}, \quad \boldsymbol{\varepsilon}^{k} \sim \mathcal{N}(0, \mathbf{I}). \tag{2}$$

The reverse process aims to recover the original data \mathbf{r}^0 from pure noise \mathbf{r}^K , where \mathbf{r}^K is sampled from the standard Gaussian distribution $\mathcal{N}(0,\mathbf{I})$. It is modeled as a Markov chain with learned Gaussian transitions: $p_{\theta}(\mathbf{r}^{k-1}|\mathbf{r}^k) = \mathcal{N}(\boldsymbol{\mu}_{\theta}(\mathbf{r}^k,k),\sigma_k^2\mathbf{I})$, where σ_k^2 is the variance, and $\boldsymbol{\mu}_{\theta}(\mathbf{r}^k,k)$ is calculated from the output of a neural network parameterized by θ (ε_{θ}). In the noise estimation framework [29], the mean is computed as

$$\boldsymbol{\mu}_{\theta}(\mathbf{r}^{k}, k) = \frac{1}{\sqrt{\alpha_{k}}} \left(\mathbf{r}^{k} - \frac{\beta_{k}}{\sqrt{1 - \bar{\alpha}_{k}}} \boldsymbol{\varepsilon}_{\theta}(\mathbf{r}^{k}, k) \right), \tag{3}$$

where $\varepsilon_{\theta}(\mathbf{r}^k, k)$ predicts the added noise at step k. The parameters θ can be learned by minimizing the MAE (denoted by $\mathcal{L}(\theta)$) between the true and predicted values:

$$\mathcal{L}(\theta) = \mathbb{E}_{k, \mathbf{r}^0, \boldsymbol{\varepsilon}} \left[\| \boldsymbol{\varepsilon}^k - \boldsymbol{\varepsilon}_{\theta}(\mathbf{r}^k, k) \|_1 \right]. \tag{4}$$

Alternatively, in the data prediction framework, we have: $\mu_{\theta}(\mathbf{r}^k, k) = \sqrt{\alpha_k}(1 - \bar{\alpha}_{k-1})\mathbf{r}^k/(1 - \bar{\alpha}_k) + \sqrt{\bar{\alpha}_{k-1}}\beta_k\mathbf{r}_{\theta}(\mathbf{r}^k, k)/(1 - \bar{\alpha}_k)$, where $\mathbf{r}_{\theta}(\mathbf{r}^k, k)$ is a network that directly estimates \mathbf{r}^0 (see [29]).

2.2 Denoising Diffusion Implicit Models (DDIMs)

Rather than formulating the inference process as a Markov process, DDIMs [28] define a non-Markovian inference process that leads to the same surrogate objective function as DDPMs: $q_{\sigma}(\mathbf{r}^{1:K}|\mathbf{r}^{0}) = q_{\sigma}(\mathbf{r}^{K}|\mathbf{r}^{0}) \prod_{k=2}^{K} q_{\sigma}(\mathbf{r}^{k-1}|\mathbf{r}^{k},\mathbf{r}^{0})$, where $q_{\sigma}(\mathbf{r}^{K}|\mathbf{r}^{0}) = \mathcal{N}(\sqrt{\bar{\alpha}_{K}}\mathbf{r}^{0},(1-\bar{\alpha}_{K})\mathbf{I})$, and for all k>1,

$$q_{\sigma}(\mathbf{r}^{k-1}|\mathbf{r}^{k},\mathbf{r}^{0}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{k-1}}\mathbf{r}^{0} + \sqrt{\frac{1-\bar{\alpha}_{k-1}-\sigma_{k}^{2}}{1-\bar{\alpha}_{k}}}\left(\mathbf{r}^{k} - \sqrt{\bar{\alpha}_{k}}\mathbf{r}^{0}\right), \ \sigma_{k}^{2}\mathbf{I}\right). \tag{5}$$

The mean is chosen such that (1) is satisfied for all k>1. In our experiments, σ_k is set to 0, and uncertainty arises purely from the diffusion endpoint \mathbf{r}^K during training and inference. To accelerate the generation process, the inference process is factored as the following non-Markovian process: $q_{\sigma,\kappa}(\mathbf{r}^{1:K}|\mathbf{r}^0) = q_{\sigma,\kappa}(\mathbf{r}^{\kappa_W}|\mathbf{r}^0) \prod_{i=1}^W q_{\sigma,\kappa}(\mathbf{r}^{\kappa_{i-1}}|\mathbf{r}^{\kappa_i},\mathbf{r}^0) \prod_{k\in\bar{\kappa}} q_{\sigma,\kappa}(\mathbf{r}^k|\mathbf{r}^0)$, where κ is a sub-sequence of $1,\ldots,K$ with length $W\leq K$ and $\kappa_W=K$, and $\bar{\kappa}=\{1,\ldots,K\}\backslash\kappa$. With the above process, we define

$$q_{\sigma,\kappa}(\mathbf{r}^k|\mathbf{r}^0) = \mathcal{N}(\sqrt{\bar{\alpha}_k}\mathbf{r}^0, (1-\bar{\alpha}_k)\mathbf{I}) \quad \forall k \in \bar{\kappa} \cup \{K\},$$

and for all $i \in \{1, ..., W\}$, the probability of $\mathbf{r}^{\kappa_{i-1}}$ conditioned on \mathbf{r}^{κ_i} and \mathbf{r}^0 is

$$q_{\sigma,\kappa}(\mathbf{r}^{\kappa_{i-1}}|\mathbf{r}^{\kappa_i},\mathbf{r}^0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{\kappa_{i-1}}}\mathbf{r}^0 + \sqrt{\frac{1-\bar{\alpha}_{\kappa_{i-1}}-\sigma_{\kappa_i}^2}{1-\bar{\alpha}_{\kappa_i}}}\left(\mathbf{r}^{\kappa_i} - \sqrt{\bar{\alpha}_{\kappa_i}}\mathbf{r}^0\right), \ \sigma_{\kappa_i}^2\mathbf{I}\right),$$

where $\kappa_0 = 0$, σ_{k_i} is again set to 0, and the coefficients are chosen such that

$$q_{\sigma,\kappa}(\mathbf{r}^{\kappa_i}|\mathbf{r}^0) = \mathcal{N}(\sqrt{\bar{\alpha}_{\kappa_i}}\mathbf{r}^0, (1 - \bar{\alpha}_{\kappa_i})\mathbf{I}) \quad \forall i \in \{1, \dots, W\}.$$
 (7)

Combining (6) and (7), we have (1) (refer to [28] for a more detailed derivation). Hence, the training objective can be the same as in DDPMs. During inference, κ can be selected to best balance between generation efficiency and diversity. Due to conflicts of naming conventions between TSF and DDPM, we use k, K, \mathbf{r} rather than t, T, \mathbf{x} in our definitions.

3 RDIT: Residual Diffusion Implicit Model with Distribution Matching

RDIT is a novel PTSF framework that incorporates distribution matching to optimize uncertainty estimates. This framework employs a two-stage procedure in both training and inference, as shown in Figure 1. In the first stage, a point-based estimator $(\mathcal{M}_{pt,\phi})$ produces point predictions of the future time series $(\hat{\mathbf{y}})$. In the second stage, a residual-conditional model $(\mathcal{M}_{res,\theta})$ (analogous to ε_{θ} in (3)), models the residuals between the ground truth (\mathbf{y}) and $\hat{\mathbf{y}}$.

We begin by describing how $\mathcal{M}_{pt,\phi}$ is used to compute essential quantities in Section 3.1, such as the residuals (r), and the standard deviation of training errors (σ_{trn}). Next, we explain how the

Point forecasting stage $\in \mathbb{R}^{M \times d}$ $\mathbf{y} \in \mathbb{R}^{M \times d}$ $\mathbf{x} \in \mathbb{R}^{N imes d}$ $\mathbf{r} \in \mathbb{R}^{M \times d}$ historic data $\mathcal{M}_{pt,\phi}$ point predictions ground truth residuals point-based estimator Residual modeling stage $\hat{\mathbf{y}}^* \in \mathbb{R}^{S \times M \times d}$ $\boldsymbol{\varepsilon} \in \mathbb{R}^{M \times d} \sim \mathcal{N}(0, \mathbf{I})$ forward/diffusion (training) final predictions sampled noise fit $\mathcal{N}(0, \sigma^2)$ $\in \mathbb{R}^{M \times d}$ $\mathfrak{M}_{res,\theta}$ residual noise estimator $\sigma_{trn} \in \mathbb{R}^{M}$ noise predictions STDEV of $\in \mathbb{R}$ optimized residuals training residuals Linear $\hat{oldsymbol{\sigma}}^* \in \mathbb{R}^{M imes d}$ estimated optimal STDEV coverage optimization J1 $\hat{\mathbf{r}}^0 \in \mathbb{R}^{S \times M \times d}$ before after reverse/denoising (inference) denoised residual

Figure 1: General scheme of this work. The point estimator $(\mathcal{M}_{pt,\phi})$ is used for predicting the point estimates $(\hat{\mathbf{y}})$, and the residual-based conditional model $(\mathcal{M}_{res,\theta})$ is used for predicting the noise $(\hat{\boldsymbol{\varepsilon}})$ from the k-th diffusion step residual (\mathbf{r}^k) while conditioning on \mathbf{x} and $\hat{\mathbf{y}}$. Two loss functions \mathcal{L}_{pt} and \mathcal{L}_{res} are optimized for training the models. During inference, $\hat{\mathbf{r}}^{\kappa_W}$ is sampled from $\mathcal{N}(0,\mathbf{I})$, and the DDIM framework is applied for accelerated denoising. The standard deviation of the residuals $(\boldsymbol{\sigma}_{trn})$ is used for normalization of the residuals. The estimated optimal standard deviation $(\hat{\boldsymbol{\sigma}}^*)$ is calculated from $\hat{\mathbf{r}}^0$ and is used to adjust the standard deviation after coverage optimization to yield the final predictions $(\hat{\mathbf{y}}^*)$.

predictions

residual-based diffusion process models the distribution of residuals using these quantities, and how accelerated inference is performed in Section 3.2. Afterward, we introduce the two distribution-matching techniques, EAE and CO, to refine the prediction distribution in Section 3.3. We then detail the neural network architecture of $\mathcal{M}_{res,\theta}$ in Section 3.4. Finally, we summarize the overall training and inference workflow and provide pseudocode in Section 3.5; further details about the problem statement of PTSF is in Appendix A.

3.1 Point-based Estimator

The point estimator $\mathfrak{M}_{pt,\phi}$ provides an initial point estimate for \mathbf{y} : $\hat{\mathbf{y}} = \mathfrak{M}_{pt,\phi}(\mathbf{x})$, where the predictions are optimized by minimizing the MAE (Appendix C) between \mathbf{y} and $\hat{\mathbf{y}}$, leading to an estimator for the conditional median [30]. Two important quantities can be calculated using $\mathfrak{M}_{pt,\phi}$: the ground truth residuals (\mathbf{r}) and standard deviation of the training residuals (σ_{trn}). Given point estimations $\hat{\mathbf{y}}$, we calculate \mathbf{r} as $\mathbf{r} = \mathbf{y} - \hat{\mathbf{y}}$ and σ_{trn} by modeling \mathbf{r} with a zero-mean Gaussian distribution using the training dataset. We obtain the standard deviation for each variate and time point in the prediction horizon, hence $\sigma_{trn} \in \mathbb{R}^{M \times d}$. Generally, σ_{trn} gets larger as the time point gets farther. This holds if the dataset lacks strong periodicity and becomes more unpredictable over time (c.f. Figure 4). However, if the periodicity is strong, the prediction uncertainty might almost stay the same (c.f. Figure 3). Therefore, σ_{trn} is used to normalize the quantities during residual-based diffusion for more consistent distributions across time and variates (Section 3.2). Using $\mathfrak{M}_{pt,\phi}$ has three advantages: First, we can leverage recent TSF models as prior knowledge for estimating the conditional median [24, 25]. Second, the other model, $\mathfrak{M}_{res,\theta}$, can be specifically designed to capture the distribution of \mathbf{r} , alleviating the burden of both point estimation and distributional modeling. Last, focusing solely on the residuals enables the application of distribution matching (Section 3.3) to enhance performance.

3.2 Residual-based Implicit Diffusion

In the second stage of our framework, we aim to model the distribution of \mathbf{r} using conditional diffusion. The starting point of diffusion is the normalized residual: $\mathbf{r}^0 = \mathbf{r}/\sigma_{trn}$. Our approach is similar to the noise prediction framework defined in (3), but additionally conditioning on the historic data (\mathbf{x}) and $\hat{\mathbf{y}}$:

$$\hat{\boldsymbol{\varepsilon}}^{0,k} = \mathcal{M}_{res,\theta}(\mathbf{r}^k, k, \mathbf{x}, \hat{\mathbf{y}}), \tag{8}$$

where \mathbf{r}^k is sampled as in (2) and the loss is defined as in (4).

In addition to the distribution matching algorithms proposed in Section 3.3, we improve upon the residual diffusion framework proposed in D3U [25] by using the *normalized* residuals, \mathbf{r}^0 , rather than raw residuals, as both the diffusion start point and denoising target. This ensures that both endpoints of the diffusion process share identical statistics, simplifying training and stabilizing the model.

To address the major disadvantage of DDPMs (Section 2.1), where significant slow-down occurs as inference time scales linearly with the number of diffusion steps K, we adopt the DDIM framework for accelerated inference (Section 2.2) and employ a subsequence κ with reduced length W < K to balance generation diversity with inference speed. σ_{κ_i} in (5) is set to $0 \ \forall i \in \{1, 2, \cdots, W\}$, and the corresponding denoising step from κ_i to κ_{i-1} is defined as

$$\hat{\mathbf{r}}^{\kappa_{i-1}} = \sqrt{\bar{\alpha}_{\kappa_{i-1}}} \hat{\mathbf{r}}^{0,\kappa_i} + \sqrt{\frac{1 - \bar{\alpha}_{\kappa_{i-1}}}{1 - \bar{\alpha}_{\kappa_i}}} \left(\hat{\mathbf{r}}^{\kappa_i} - \sqrt{\bar{\alpha}_{\kappa_i}} \hat{\mathbf{r}}^{0,\kappa_i} \right), \tag{9}$$

where $\hat{\mathbf{r}}^{\kappa_{i-1}}$ is the prediction at step $k=\kappa_{i-1}$, $\hat{\mathbf{r}}^{\kappa_0}=\hat{\mathbf{r}}^0=\hat{\mathbf{r}}/\sigma_{trn}$ is the generation endpoint, $\hat{\mathbf{r}}^{0,\kappa_i}$ is the normalized predicted ground truth using (3) and (8) with $k=\kappa_i$, and $\bar{\alpha}_{\kappa_i}$ is calculated using (1) with $k=\kappa_i$. Since the denoising endpoint $\hat{\mathbf{r}}^0$ is already normalized, we constrain the intermediate outcomes to have zero mean and unit variance; thus, only the shape of the distribution changes during denoising. While increasing the number of diffusion steps κ can improve sample fidelity, too many steps may cause target deviation or artifacts, leading to a more erroneous mean. In our experiments, a κ with $W\approx 10$ suffices for effective diffusion, as $\mathcal{M}_{pt,\phi}$ already provides a reliable median estimate. Additional experiments show that, in general, the denoising process gradually improves the CRPS at each denoising step (Appendix L).

3.3 Distribution Matching

In this section, we introduce the EAE and CO algorithms to address potential drawbacks that can lead to suboptimal performance in current PTSF methods.

3.3.1 Error-aware Expansion (EAE)

An important metric used for evaluating the performance of probabilistic forecasts is the CRPS [31] (Appendix C). During training, minimizing MAE optimizes the accuracy of point estimates, but it disregards the uncertainty or spread of the predictions. This can lead to overconfident predictions. The CRPS, on the other hand, rewards models that provide a well-calibrated probabilistic distribution. A model that minimizes MAE might produce predictions that are tightly clustered around the median or mean, potentially leading to poorer probabilistic forecasts when evaluated with CRPS. One method to mitigate this issue is to directly minimize the CRPS during training [27]. However, multiple values for the same ground truth must be sampled in order to accurately calculate the CRPS, resulting in slower training. We thus introduce the EAE method to modify the original predictions and match the optimal distribution that minimizes the CRPS. This becomes feasible given the following theorem:

Theorem 3.1. Given ground truth y and predictions \hat{y} , where $\hat{y} \sim \mathcal{N}(0, \sigma^2)$, where σ is the standard deviation, let $F_{\hat{y},\sigma}$ be the CDF of \hat{y} , then the σ that minimizes the CRPS (σ^*) is

$$\sigma^{\star} = \operatorname*{arg\,min}_{\sigma>0} \mathrm{CRPS}(F_{\hat{y},\sigma},y) = \frac{|y|}{\sqrt{\ln 2}}.$$

Proof. See Appendix E.

Corollary 3.2. Given y and $\hat{y} \sim \mathcal{N}(\mu, \sigma^2)$, the σ that minimizes the CRPS (σ^*) is

$$\sigma^* = \operatorname*{arg\,min}_{\sigma>0} \operatorname{CRPS}(F_{\hat{y},\sigma}, y) = \frac{|y-\mu|}{\sqrt{\ln 2}}.\tag{10}$$

Based on the above corollary, we can calculate the standard deviation that minimizes the CRPS given the ground truth and normally distributed predictions with a fixed mean. However, during inference, the absolute error $|y - \mu|$ is unknown. In our framework, the final prediction is $z = \hat{y} + \hat{r}$. Since $\mathcal{M}_{pt,\phi}$ is trained using the MAE loss, we can assume $\mathbb{E}[\hat{r}] \to 0$. Therefore, we get

$$z = \hat{y} + \hat{r} \sim \mathcal{N}(\mu, \sigma^2) = \mathcal{N}(\hat{y}, \sigma^2) \quad \Longrightarrow \quad |y - \mu| = |y - \hat{y}| = |r|. \tag{11}$$

This shows that one can compute σ^* from the magnitude of the ground-truth residual |r|. If the residual model effectively captures uncertainty, then |r| is likely positively correlated with $|\hat{r}|$, so we approximate $|r| \approx \alpha \mathbb{E}[|\hat{r}|]$, where α is the linear coefficient between |r| and $\mathbb{E}[|\hat{r}|]$. Combining this approximation with (10) and (11), we obtain

$$\sigma^{\star} \approx \frac{\alpha \, \mathbb{E}[|\hat{r}|]}{\sqrt{\ln 2}}.\tag{12}$$

(12) suggests that the optimal standard deviation is proportional to the expected magnitude of the residuals. This is intuitive: greater residual magnitude implies higher uncertainty, which should be reflected by a larger σ^* . By choosing an appropriate α and estimating σ^* from \hat{r} , we can minimize the CRPS even when the model is trained with the MAE objective. After computing σ^* , we expand or shrink the predictions to match this target variance. We apply this to each $\hat{\mathbf{r}}_{ij} \in \mathbb{R}^S$ in $\hat{\mathbf{r}} \in \mathbb{R}^{S \times M \times d}$, where $i \leq M, j \leq d$, and each $\hat{\mathbf{r}}_{ij}$ is treated as an empirical Gaussian distribution. Experimental results demonstrate the effectiveness of EAE by comparing it to the use of a fixed expansion factor across all time points (Appendix J). Visualizations are also shown in Appendix M.

3.3.2 Coverage Optimization (CO)

Modeling the distribution of the residuals is challenging and trained models might fail to capture the true distribution of the data due to model bias [32]. As a result, a predictor may not have sufficient inductive bias to assign the correct probability to every credible interval. Moreover, due to relatively larger noise in the residuals, overfitting may occur, which is exacerbated by concept drift. In this section, we present the CO algorithm for calibrating the prediction intervals of time series data, achieving an effect similar to that of Kuleshov et al. [33] but suitable for empirical distributions.

Since the mismatch between the true and predictive distributions can be effectively captured by the Prediction Interval Coverage Probability (PICP) (see detailed definition in Appendix C), we propose to calibrate the predictive distribution by splitting it into several prediction intervals and finding suitable expansion factors for each interval. Specifically, we define target prediction quantiles to match. Let $\gamma = \{\gamma_0, \gamma_1, \cdots, \gamma_l\}$ be a collection of quantiles of the predictive distribution, where $0 \le \gamma_i < \gamma_{i+1} < 1, \forall \ 0 \le i < l$. During the i-th step (i < l), we adjust the distribution width of all $\hat{r} \in (-\infty, F_{\hat{r}}^{-1}(\frac{1-\gamma_i}{2})] \cup [F_{\hat{r}}^{-1}(\frac{1+\gamma_i}{2}), \infty)$ such that $\mathrm{PICP}(\gamma_{i+1}, r, F_{\hat{r}}) = \gamma_{i+1}$. Since $\hat{r} \in (F_{\hat{r}}^{-1}(\frac{1-\gamma_i}{2}), F_{\hat{r}}^{-1}(\frac{1+\gamma_i}{2}))$ is fixed, it follows that the $\mathrm{PICP}(\gamma_j, r, F_{\hat{r}})$ for all $j \le i$ remain unchanged during the i-th step. By incrementing i and adjusting the distribution width, we can calibrate the whole distribution and match the coverage probabilities at any desired prediction interval for the calibration dataset. Detailed visualizations are shown in Appendix I.

Accordingly, for all i < l, we implement binary search and calculate corresponding expansion factors $\lambda = \{\lambda_0, \lambda_1, \cdots, \lambda_l - 1\}$ for the adjustment, such that any \hat{r} within range is adjusted as:

$$\hat{r}_{CO} \leftarrow \begin{cases} F_{\hat{r}}^{-1}(\frac{1-\gamma_i}{2}) - \lambda_i (F_{\hat{r}}^{-1}(\frac{1-\gamma_i}{2}) - \hat{r}), & \hat{r} \in (-\infty, F_{\hat{r}}^{-1}(\frac{1-\gamma_i}{2})]; \\ F_{\hat{r}}^{-1}(\frac{1+\gamma_i}{2}) + \lambda_i (\hat{r} - F_{\hat{r}}^{-1}(\frac{1+\gamma_i}{2})), & \hat{r} \in [F_{\hat{r}}^{-1}(\frac{1+\gamma_i}{2}), \infty). \end{cases}$$
(13)

We use the validation dataset to generate λ and apply them during testing in practice. Similar techniques for calibrating using the validation dataset are proposed in several time series analysis literature [33, 34, 35, 36]. Our method stands out as a simple yet effective approach for empirical distributions, without requiring any assumptions about the underlying distribution.

3.4 Diffusion Model Network Architecture

In this section, we provide detailed steps as to how the neural network model $\mathcal{M}_{res,\theta}$ in (8) denoises \mathbf{r}^k and provides an estimation of the ground truth noise $(\hat{\boldsymbol{\varepsilon}}^{0,k})$. All the concatenation operations are performed sequentially on the first dimension. In particular, given $\mathbf{a} \in \mathbb{R}^{d_a \times D}$ and $\mathbf{b} \in \mathbb{R}^{d_b \times D}$, we have $\mathrm{concat}(\mathbf{a},\mathbf{b}): \mathbb{R}^{d_a \times D} \times \mathbb{R}^{d_b \times D} \to \mathbb{R}^{(d_a+d_b) \times D}$. Throughout this section, $\mathbf{z}_i, \forall i \in \{1,2,\cdots,6\}$ stands for intermediate representations calculated in the network. We concatenate \mathbf{x} and $\hat{\mathbf{y}}$, individually process each variate by standard normalization, then concatenate the result with \mathbf{r}^k such that $\mathbf{z}_1 = \mathrm{concat}(\mathbf{x},\hat{\mathbf{y}}) \in \mathbb{R}^{(N+M) \times d}$ and $\mathbf{z}_2 = \mathrm{concat}(\tilde{\mathbf{z}}_1,\mathbf{r}^k) \in \mathbb{R}^{(N+2M) \times d}$. The resulting embedding \mathbf{z}_2 is then transposed, passed through a fully connected layer, and then passed through a GELU activation function [37], yielding \mathbf{z}_3 . We then embed the diffusion step k with two fully connected layers and concatenate it to \mathbf{z}_3 , yielding \mathbf{z}_4 :

$$\mathbf{z}_3 = \mathtt{GELU}(\mathtt{Linear}(\mathbf{z}_2^\top)) \in \mathbb{R}^{d \times H}, \quad \mathbf{z}_4 = \mathtt{concat}(\mathbf{z}_3, \mathtt{embed}(k)) \in \mathbb{R}^{(d+d_k) \times H}.$$
 (14)

To this point, all variates are independently processed. To further capture the inter-variate dependencies along with time-dependent relationships, we incorporate bidirectional Mamba layers as

$$\mathbf{z}_5 = [\mathtt{Mamba}(\mathbf{z}_4^{ op}) + \mathtt{Mamba}(\mathtt{Flip}(\mathbf{z}_4^{ op}))]^{ op} \in \mathbb{R}^{(d+d_k) imes H}.$$

The usage of a bidirectional Mamba layer stems from the fact that Mamba's selection mechanism can only incorporate antecedent time points [21]. Instead of applying attention in the variate dimension [22], we apply attention in the time dimension, which we found to be more effective.

After incorporating the Mamba layer, the output \mathbf{z}_5 is projected back to its original length M, and the first d variates are extracted, yielding $\mathbf{z}_6 = [\mathtt{Linear}(\mathbf{z}_5)^\top]_{0:d} \in \mathbb{R}^{M \times d}$. Building on top of the assumption that \mathbf{r}^0 is normalized yielding a mean of 0 and variance of \mathbf{I} , finally, we add a portion of \mathbf{r}^k to \mathbf{z}_6 as the final output: $\hat{\boldsymbol{\varepsilon}}^{0,k} = \mathbf{z}_6 + \mathbf{r}^k k/K \in \mathbb{R}^{M \times d}$. By calculation, the correlation between \mathbf{r}^k and $\boldsymbol{\varepsilon}^k$ component-wise is

$$\rho_{r^k,\varepsilon^k} = \frac{\operatorname{cov}(r^k,\varepsilon^k)}{\sqrt{\operatorname{var}(r^k)\operatorname{var}(\varepsilon^k)}} = \mathbb{E}[r^k\varepsilon^k] = \sqrt{1-\bar{\alpha}_k}\mathbb{E}[(\varepsilon^k)^2] = \sqrt{1-\bar{\alpha}_k},\tag{15}$$

where the third equality holds because $r^0 \perp \!\!\! \perp \varepsilon^k$. From (1), we have $\rho_{r^k,\varepsilon^k} \to 1$ as $k \to K$. Therefore, adding a scalable portion of \mathbf{r}^k to the final output increases the inductive bias of the model.

3.5 Training and Inference

The training algorithm is shown in Algorithm 1. For each x, we sample a k and ε^k , and minimize an aggregated loss function for simultaneously optimizing $\mathcal{M}_{pt,\phi}$ and $\mathcal{M}_{res,\theta}$, i.e.,

$$\min_{\phi,\theta} \mathcal{L}(\phi,\theta) = \min_{\phi,\theta} \mathbb{E}_{\mathbf{x},\pmb{\varepsilon},k}[\mathcal{L}_{pt}(\phi) + \mathcal{L}_{res}(\theta)],$$

where $\mathcal{L}_{pt}(\phi) = \|\mathbf{y} - \mathcal{M}_{pt,\phi}(\mathbf{x})\|_1$ is the L_1 -loss between the ground truth future values and the point-based predictions, and $\mathcal{L}_{res}(\theta) = \|\boldsymbol{\varepsilon}^k - \mathcal{M}_{res,\theta}(\mathbf{r}^k,k,\mathbf{x},\hat{\mathbf{y}})\|_1$ is the L_1 -loss between the ground truth residuals and the denoised residuals.

The inference algorithm is shown in Algorithm 2. For each \mathbf{x} , we generate a $\hat{\mathbf{y}}$ using $\mathcal{M}_{pt,\phi}$. During the residual denoising stage, both \mathbf{x} and $\hat{\mathbf{y}}$ are used as the conditioning variable. We generate $\mathbf{r}^K = \boldsymbol{\varepsilon}^{\kappa_W} \sim \mathcal{N}(0,\mathbf{I})$ and apply (8) and (9) repeatedly until $\hat{\mathbf{r}}^0$ is obtained, then de-normalize it using $\boldsymbol{\sigma}_{trn}$. After distribution matching (Section 3.3), the final prediction is

$$\hat{\mathbf{y}}^* = \hat{\mathbf{y}} + \mathbb{E}[\hat{\mathbf{r}}_{CO}] + \lambda_{EAE}(\hat{\mathbf{r}}_{CO} - \mathbb{E}[\hat{\mathbf{r}}_{CO}]), \tag{16}$$

where the expand factor for EAE, $\lambda_{EAE} = \sigma^*/\sigma_{\hat{\mathbf{r}}_{CO}}$, corresponds to the ratio that the distribution needs to be expanded to match the optimal distribution. If no distribution matching is applied, then

Algorithm 1 Training

$\begin{array}{l} \textbf{repeat} \\ \textbf{Sample } \textbf{x} \textbf{ and } \textbf{y} \textbf{ from the training set} \\ \hat{\textbf{y}} \leftarrow \mathcal{M}_{pt,\phi}(\textbf{x}) \\ \textbf{r}^0 \leftarrow (\textbf{y} - \hat{\textbf{y}})/\sigma_{trn} \\ k \sim \textbf{Uniform}(\{1,2,...,K\}) \\ \varepsilon \sim \mathcal{N}(0,\textbf{I}) \\ \textbf{r}^k \leftarrow \sqrt{\bar{\alpha}_k} \textbf{r}^0 + \sqrt{1-\bar{\alpha}_k} \varepsilon \\ \hat{\varepsilon}^{0,k} \leftarrow \mathcal{M}_{res,\theta}(\textbf{r}^k,k,\textbf{x},\hat{\textbf{y}}) \\ \textbf{Calculate } \mathcal{L}_{pt}(\phi) \textbf{ and } \mathcal{L}_{res}(\theta) \\ \textbf{Do gradient descent on } \nabla_{\theta}\mathcal{L}_{res} \textbf{ and } \nabla_{\phi}\mathcal{L}_{pt} \\ \textbf{until converged} \end{array}$

Algorithm 2 Inference

```
\begin{split} & \textbf{Input: } \mathbf{x}, \boldsymbol{\sigma}_{trn} \\ & \hat{\mathbf{y}} \leftarrow \mathcal{M}_{pt,\phi}(\mathbf{x}) \\ & \mathbf{r}^K \leftarrow \mathcal{N}(0, \mathbf{I}) \\ & \textbf{for } i = W \textbf{ to } 1 \textbf{ do} \\ & \hat{\boldsymbol{\varepsilon}}^{0,\kappa_i} \leftarrow \mathcal{M}_{res,\theta}(\hat{\mathbf{r}}^{\kappa_i}, \kappa_i, \mathbf{x}, \hat{\mathbf{y}}) \\ & \text{Calculate } \hat{\mathbf{r}}^{\kappa_{i-1}} \text{ using } (9) \\ & \textbf{end for} \\ & \hat{\mathbf{r}} \leftarrow \hat{\mathbf{r}}^0 \boldsymbol{\sigma}_{trn} \\ & \hat{\mathbf{r}}_{CO} \leftarrow \text{Adjust } \hat{\mathbf{r}} \text{ using } (13) \\ & \boldsymbol{\lambda}_{EAE} \leftarrow \alpha \mathbb{E}[|\hat{\mathbf{r}}_{CO}|]/\boldsymbol{\sigma}_{\hat{\mathbf{r}}_{CO}} \sqrt{ln2} \\ & \hat{\mathbf{y}}^* \leftarrow \hat{\mathbf{y}} + \mathbb{E}[\hat{\mathbf{r}}_{CO}] + \boldsymbol{\lambda}_{EAE}(\hat{\mathbf{r}}_{CO} - \mathbb{E}[\hat{\mathbf{r}}_{CO}]) \\ & \mathbf{return } \hat{\mathbf{y}}^* \end{split}
```

4 Experiments

4.1 Datasets

Experiments are conducted on eight widely used public datasets [21]. The Traffic dataset¹ contains hourly road occupancy rates collected from 862 sensors, recorded between January 2015 and December 2016. The Weather dataset² contains 21 meteorological indicators collected every 10 minutes during 2020. The Electricity dataset³ records the hourly electricity consumption of 321 clients from 2012 to 2014. The Exchange dataset records daily exchange rates of eight countries from 1990 to 2016 [38]. The Solar dataset records the solar power production of 137 PV plants, sampled every 10 minutes during 2006 [38]. The ETT dataset contains data on load and oil temperature collected from electricity transformers between July 2016 and July 2018. We use the subsets ETTh1, recorded hourly, and ETTm1 and ETTm2, recorded every 15 minutes, which capture seven transformer-related factors [39]. The statistics of the datasets are summarized in Table 1.

Table 1: Datasets used in this work.

Dataset	Traffic		Electricity	Exchange	ETTm1	ETTm2	ETTh1	Solar
Variates (d)	862	21	321	8	7	7	7	137
Timesteps	17,544	52,696	26,304	7,588	69,680	69,680	17,420	52,560
Frequency	1 hour	10 min	1 hour	1 day	15 min	15 min	1 hour	10 min

4.2 Baselines

Three types of baselines, comprising a total of ten methods, are compared to RDIT:

- (i) Point-based TSF models [17, 21, 22, 40, 41]. These models provide point estimates (\hat{y}) and typically focus on the MSE and MAE. Directly using \hat{y} and calculating their CRPS will result in a value identical to the MAE. However, this is not a fair comparison, as shown in Appendix E. We thus *reinforce* the TSF models by adding $\mathcal{N}(0, \sigma_{trn}^2)$ to \hat{y} .
- (ii) Diffusion-based PTSF models [24, 42, 43, 25]. These models focus on generating a distribution that covers as much portion of the ground truth as possible, hence, the CRPS and PICP can be used naturally.
- (iii) Foundation models [44]. These large-scale pretrained models support zero- and few-shot forecasting and deliver probabilistic predictions with calibrated uncertainties.

4.3 Main Results

We incorporate TimeFilter [41] as the point estimator $\mathcal{M}_{pt,\phi}$ and the bidirectional Mamba-based network as the diffusion/denoising model $\mathcal{M}_{res,\theta}$ in this section. Table 2 shows the CRPS (21) and

¹https://pems.dot.ca.gov/

²https://www.bgc-jena.mpg.de/wetter/

https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

PICP distances (23) of our method compared to the baselines (full results in Appendix G). For each metric, RDIT excelled on all but one of the eight datasets, indicating strong adaptability to diverse data distributions and effective capture of both long-term and short-term temporal dependencies. One particularly surprising result is the performance of point-based TSF methods augmented with zero-mean Gaussians: CRPS achieved the second-best score on six of eight datasets, and PICP distance achieved the second-best score on five of eight datasets, often outperforming state-of-the-art PTSF methods. This establishes these enhanced point estimators as strong baselines and demonstrates that Gaussian augmentation is effective for uncertainty modeling. Moreover, these findings highlight that a high-quality point estimator is critical for producing accurate probabilistic forecasts.

We compare the MAE and MSE of our method and the PTSF models in Table 3 (full results in Appendix K). To ensure a fair comparison of the point metrics (cf. Appendix D), we first collapse the predictive distributions onto their means and use these as point estimates for evaluation. RDIT still achieves state-of-the-art performance in terms of both MAE and MSE. This demonstrates that RDIT excels not only in PTSF tasks but also in the degenerated point forecasting setting. On the other hand, the results for zero-shot prediction using Chronos suggest that it remains challenging for such models to outperform state-of-the-art supervised learning methods. Visualizations comparing the ground truth and the prediction intervals are shown in Appendix H.

Table 2: CRPS and PICP distance of different algorithms for PTSF for eight datasets averaged across prediction lengths 24, 48, and 96. **Bold**: best (lowest) value; underlined: second to best.

	Dataset	T	raffic	We	eather	Ele	ctricity	Exc	hange	E7	lTm1	ET	Tm2	E	ΓTh1	So	olar
Type	Method	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis
LLM	Chronos	0.358	0.929	0.156	0.920	0.254	1.059	0.124	0.679	0.410	0.818	0.212	0.921	0.342	0.802	0.473	0.978
-	TimeDiff iTransformer	0.349	$\frac{0.114}{0.352}$	0.545	0.465 0.406	0.276	$\frac{0.067}{0.180}$	0.135	0.152 0.177	0.324	0.141 0.142	0.197	0.158 0.179	0.298	0.218 0.178	0.272	0.210
TSF	SMamba	0.233	0.351	0.184	0.410	0.172	0.337	0.126	0.171	0.258	0.138	0.183	0.182	0.294	0.197	0.207	0.318
	PatchTST TimeFilter	0.248	0.377	0.178	0.458 0.414	$\frac{0.162}{0.162}$	0.200	0.120 0.112	$\frac{0.142}{0.161}$	$\frac{0.240}{0.246}$	0.162	$\frac{0.175}{0.178}$	0.185 0.189	0.299	0.189	0.179 0.186	0.342 0.332
	SSSD	0.517	1.639	10.357	1.264	0.549	1.563	1.001	1.783	10.577	1.133	1.011	1.697	0.623	0.925	0.436	1.474
	Tactis-2	0.443	0.780	0.168	0.447	0.335	1.054	0.130	0.182	0.279	0.262	0.292	0.681	0.508	0.848	0.216	1.272
PTSF	TMDM D3U	0.237 0.213	0.172 0.113	0.160 0.149	0.486 0.180	0.211	0.143 0.108	0.232 0.129	0.203 0.371	0.330 0.248	0.267 0.167	0.219 0.193	0.507 0.145	0.368	0.336 0.523	0.201 0.150	0.293 0.129
	RDIT	0.200	0.147	0.132	0.158	0.161	0.065	0.111	0.063	0.239	0.034	0.174	0.095	0.287	0.124	0.178	0.047

Table 3: MAE and MSE of different algorithms for PTSF for eight datasets averaged across prediction lengths 24, 48, and 96. **Bold**: best (lowest) value; <u>underlined</u>: second to best.

Dataset	Tra	ffic	Wea	ther	Elect	ricity	Exch	ange	ETT	Γm1	ET1	Γm2	ET	Th1	Sol	ar
Method	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Chronos	0.485	0.906	0.230	0.262	0.339	0.341	0.197	0.181	0.595	0.978	0.328	0.344	0.487	0.626	0.644	1.175
SSSD	0.571	0.879	0.444	0.387	0.643	0.662	1.118	1.928	0.709	0.919	1.141	2.325	0.774	1.114	0.574	0.585
Tactis-2	0.564	0.846	0.220	0.167	0.411	0.365	0.174	0.068	0.372	0.372	0.360	0.293	0.625	0.973	0.270	0.220
TMDM	0.297	0.330	0.195	0.170	0.275	0.185	0.318	0.181	0.420	0.455	0.272	0.208	0.473	0.529	0.251	0.204
D3U	0.281	0.496	0.196	0.145	0.245	0.151	0.171	0.062	0.337	0.274		0.153	0.415	0.389	0.222	0.163
RDIT	0.224	0.307	0.159	0.116	0.210	0.112	0.150	0.051	0.325	0.271	0.224	0.129	0.377	0.336	0.203	0.153

4.4 Ablation Study

Table 4 shows the experimental results by incrementally applying different components of our method and comparing their performance. We start with the point estimates directly generated from $\mathcal{M}_{pt,\phi}$. Since there is no predictive distribution, the PICP distance cannot be measured, and the CRPS is equal to the MAE (Appendix E). The 1-step denoise row corresponds to setting $\kappa = \{K\}$, where only one denoising step is taken to predict the residuals. After applying DDIM with ten denoising steps ($\kappa = \{K/10, 2K/10, \cdots, K\}$), we observe improvements in CRPS for most datasets; however, the PICP distance does not consistently improve, indicating that gradual denoising does not ensure the coverage is accurately captured. Adding EAE leads to slight improvements in CRPS, and by further incorporating CO, we achieve the best overall performance across both metrics. These results suggest that each component of our framework contributes to enhancing predictive performance. Additional experiments using SMamba [21] as the point estimator also showed that applying the RDIT framework results in additive improvement (Appendix F).

Table 4: CRPS and PICP distance of ablation study results averaged across prediction lengths 24, 48, 96, 192, 336, and 720. **Bold**: best (lowest) value; underlined: second to best.

Dataset	W	eather	Exc	change	El	ΓTm1	E	ΓTm2	E	ΓTh1	So	olar
Features	CRPS	PICP dis	CRPS	PICP dis	CRPS	PICP dis						
Point estimate (\hat{y})	0.225	N/A	0.311	N/A	0.367	N/A	0.289	N/A	0.412	N/A	0.235	N/A
1-step denoise	0.209	0.373	0.287	0.137	0.339	0.179	0.261	0.226	0.389	0.221	0.207	0.293
DDIM	0.200	0.233	0.225	0.141	0.278	0.194	0.226	0.201	0.314	0.225	0.209	0.368
DDIM+EAE	0.196	0.303	0.225	0.123	0.277	0.169	0.225	0.179	0.313	0.201	0.209	0.419
DDIM+EAE+CO	0.186	0.148	0.223	0.059	0.275	0.042	0.226	$\overline{0.087}$	0.314	$\overline{0.110}$	0.198	0.079

5 Conclusion

In conclusion, we present RDIT, a novel framework integrating diffusion processes for probabilistic time series forecasting. It consists of a plug-and-play point-based model that estimates the conditional median, and a residual-based model that estimates the distribution of residuals based on this median, smoothly balancing the workload between the two models. We prove that the CRPS can be further optimized for predictions with a Gaussian distribution and derive the EAE and CO algorithms for distribution matching. An important future research is to extend the current approach to arbitrary predictive distributions beyond the Gaussian. Our results show that RDIT is highly efficient due to the usage of DDIMs and superior across datasets from different fields and a wide range of prediction lengths. By modeling uncertainty effectively, we can potentially improve the modeling of market risk for quantitative trading, accelerate decision-making in critical domains such as healthcare, weather forecasting, and manufacturing, and contribute to enhanced AI reliability.

References

- [1] Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing*, 90:106181, 2020
- [2] Mohammad Amin Morid, Olivia R. Liu Sheng, and Joseph Dunbar. Time series prediction using deep learning methods in healthcare. *ACM Trans. Manage. Inf. Syst.*, 14(1), January 2023.
- [3] Syeda Sitara Wishal Fatima and Afshin Rahimi. A review of time-series forecasting algorithms for industrial manufacturing systems. *Machines*, 12(6), 2024.
- [4] Alina Bărbulescu. Studies on time series applications in environmental sciences. In *Intelligent Systems Reference Library*, 2016.
- [5] Linkai Wang, Jing Chen, Wei Wang, Ruizhuo Song, Zhaochong Zhang, and Guowei Yang. Review of time series traffic forecasting methods. In 2022 4th International Conference on Control and Robotics (ICCR), pages 1–5, 2022.
- [6] David Karl. Forecasting e-commerce consumer returns: a systematic literature review. *Management Review Quarterly*, May 2024.
- [7] Shengzhong Mao, Chaoli Zhang, Yichi Song, Jindong Wang, Xiao-Jun Zeng, Zenglin Xu, and Qingsong Wen. Time series analysis for education: Methods, applications, and future directions. arXiv preprint arXiv:2408.13960, 2024.
- [8] John A Miller, Mohammed Aldosari, Farah Saeed, Nasid Habib Barna, Subas Rana, I Budak Arpinar, and Ninghao Liu. A survey of deep learning and foundation models for time series forecasting. arXiv preprint arXiv:2401.13912, 2024.
- [9] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 379(2194):20200209, 2021.
- [10] José F. Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. Deep learning for time series forecasting: A survey. *Big Data*, 9(1):3–21, 2021. PMID: 33275484.
- [11] Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(Volume 1, 2014):125–151, 2014.
- [12] Ioannis K. Bazionis and Pavlos S. Georgilakis. Review of deterministic and probabilistic wind power forecasting: Models, methods, and future research. *Electricity*, 2(1):13–47, 2021.

- [13] Hristos Tyralis and Georgia Papacharalampous. A review of probabilistic forecasting and prediction with machine learning. *arXiv* preprint arXiv:2209.08307, 2022.
- [14] Lequan Lin, Zhengkun Li, Ruikun Li, Xuliang Li, and Junbin Gao. Diffusion models for time-series applications: a survey. Frontiers of Information Technology & Electronic Engineering, 25(1):19–41, Jan 2024.
- [15] Marcel Kollovieh, Abdul Fatir Ansari, Michael Bohlke-Schneider, Jasper Zschiegner, Hao Wang, and Yuyang Bernie Wang. Predict, refine, synthesize: Self-guiding diffusion models for probabilistic time series forecasting. Advances in Neural Information Processing Systems, 36:28341–28364, 2023.
- [16] Caspar Meijer and Lydia Y Chen. The rise of diffusion models in time-series forecasting. arXiv preprint arXiv:2401.03006, 2024.
- [17] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv* preprint arXiv:2211.14730, 2022.
- [18] Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023.
- [19] William Toner and Luke Darlow. An analysis of linear time series forecasting models. *arXiv preprint* arXiv:2403.14587, 2024.
- [20] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf. Multivariate probabilistic time series forecasting via conditioned normalizing flows. arXiv preprint arXiv:2002.06103, 2020.
- [21] Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025.
- [22] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. arXiv preprint arXiv:2310.06625, 2023.
- [23] Zongjiang Shang, Ling Chen, Dongliang Cui, et al. Ada-mshyper: Adaptive multi-scale hypergraph transformer for time series forecasting. *arXiv* preprint arXiv:2410.23992, 2024.
- [24] Yuxin Li, Wenchao Chen, Xinyue Hu, Bo Chen, baolin sun, and Mingyuan Zhou. Transformer-modulated diffusion models for probabilistic multivariate time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.
- [25] Qi Li, Zhenyu Zhang, Lei Yao, Zhaoxia Li, Tianyi Zhong, and Yong Zhang. Diffusion-based decoupled deterministic and uncertain framework for probabilistic multivariate time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [26] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. Advances in Neural Information Processing Systems, 35:5775–5787, 2022.
- [27] Vincent Zhihao Zheng and Lijun Sun. Mvg-crps: A robust loss function for multivariate probabilistic forecasting. arXiv preprint arXiv:2410.09133, 2024.
- [28] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint* arXiv:2010.02502, 2020.
- [29] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- [30] Joram Soch, The Book of Statistical Proofs, Karahan Sarıtaş, Maja, Pietro Monticone, Thomas J Faulkenberry, Osvaldo A Martin, Alex Kipnis, Salvador Balkus, Ifkdlfdlk, Carsten Allefeld, Heiner Atze, Adam Knapp, Ciarán D McInerney, Lo4ding, Valeriu Ohan, amvosk, and maxgrozo. StatProofBook/StatProofBook.github.io: StatProofBook 2024, 2025.
- [31] James E. Matheson and Robert L. Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22(10):1087–1096, 1976.
- [32] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

- [33] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International conference on machine learning*, pages 2796–2804. PMLR, 2018.
- [34] Chih-Yu (Andrew) Lai, Fan-Keng Sun, Zhengqi Gao, Jeffrey H Lang, and Duane Boning. Nominality score conditioned time series anomaly detection by point/sequential reconstruction. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 76637–76655. Curran Associates, Inc., 2023.
- [35] Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Conformal prediction with temporal quantile adjustments. *Advances in Neural Information Processing Systems*, 35:31017–31030, 2022.
- [36] Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. Advances in neural information processing systems, 32, 2019.
- [37] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415, 2016.
- [38] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.
- [39] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.
- [40] Lifeng Shen and James Kwok. Non-autoregressive conditional diffusion models for time series prediction. In *International Conference on Machine Learning*, pages 31016–31029. PMLR, 2023.
- [41] Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. Timefilter: Patch-specific spatial-temporal graph filtration for time series forecasting. *arXiv* preprint arXiv:2501.13041, 2025.
- [42] Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv* preprint arXiv:2208.09399, 2022.
- [43] Arjun Ashok, Étienne Marcotte, Valentina Zantedeschi, Nicolas Chapados, and Alexandre Drouin. Tactis-2: Better, faster, simpler attentional copulas for multivariate time series. arXiv preprint arXiv:2310.01327, 2023.
- [44] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.
- [45] Zhiqing Sun and Yiming Yang. Difusco: Graph-based diffusion solvers for combinatorial optimization. *Advances in neural information processing systems*, 36:3706–3731, 2023.

A Probabilistic Time Series Forecasting

Time Series Forecasting (TSF) aims to predict future values of multiple interrelated variables based on their historical observations. Let $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ denote the history of the time series, where each $\mathbf{x}_t \in \mathbb{R}^d$ represents the values of d variables at time step t, and N is the length of the historical data. The prediction target is a sequence $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_M\}$, where each $\mathbf{y}_t \in \mathbb{R}^d$ corresponds to the future values we aim to forecast over a horizon of length M. The TSF problem can be formulated mathematically as finding a function f such that

$$\mathbf{y} = f(\mathbf{x}),\tag{17}$$

where $f: \mathbb{R}^{N \times d} \to \mathbb{R}^{M \times d}$ maps the historical data \mathbf{x} to the future observations \mathbf{y} .

Probabilistic Time Series Forecasting (PTSF) aims to predict the future probability distributions of multiple interrelated variables based on their historical observations, where the prediction targets are defined as $\mathbf{y} = \{p(\mathbf{y}_1), p(\mathbf{y}_2), \dots, p(\mathbf{y}_M)\}$. The PTSF problem can be formulated mathematically as modeling the conditional distribution of the future observations given the history. Specifically, the goal is to find a function f such that

$$p(\mathbf{y}|\mathbf{x}) = f(\mathbf{x}),\tag{18}$$

where $f: \mathbb{R}^{N \times d} \to \mathcal{P}(\mathbb{R}^{M \times d})$ maps the historical data \mathbf{x} to the joint probability distribution over the future observations \mathbf{y} . Here, $\mathcal{P}(\mathbb{R}^{M \times d})$ denotes the space of probability distributions over $\mathbb{R}^{M \times d}$. This allows for the quantification of uncertainty in the predictions, which is crucial for risk assessment and decision-making in various applications.

B Implementation Details

The parameters in our setting are largely inspired by [21, 41, 45]. For all the point estimators (SMamba and TimeFilter), we use the same settings as in the original literature. All experiments were implemented using the PyTorch framework and conducted on an NVIDIA RTX 3090 24GB GPU. The train/validation/test split was maintained at 70/10/20 percent across all experiments. PyTorch Lightning library was employed to streamline the code implementation, with the number of workers set to either 16 or 32. The Adam optimizer with weight decay was chosen for optimization, and early stopping was applied during the training of $\mathcal{M}_{pt,\phi}$ to enhance stability. The detailed parameters for training/testing is shown in Table 5. The superscript of parameter values is the corresponding prediction length.

Table 5: Implementation settings in this work.

	I	I				Datase	et		
Parameter	Naming	Traffic	Weather	Electricity	Exchange	ETTm1	ETTm2	ETTh1	Solar
				General					
activation function		GELU	GELU	GELU	GELU	GELU	GELU	GELU	GELU
train batch size	batch_size	16	32	$32^{\leq 48}$, $16^{\geq 96}$	32	32	32	32	32
validation/test batch size	test_batch_size	1	1	1	1	1	1	1	1
# of training epochs	num_epochs	100	100	100	100	100	100	100	100
			D	DIM parameters					
# of samples for (S)	samples	50	100	100	100	100	100	100	100
# of diffusion steps (K)	diffusion_steps	1000	1000	1000	1000	1000	1000	1000	1000
length of κ	inference_diffusion_steps	10	10	10	10	10	10	10	10
inference schedule	inference_schedule	cosine	cosine	cosine	cosine	cosine	cosine	cosine	cosine
β schedule	beta_schedule	linear	linear	linear	linear	linear	linear	linear	linear
			Noise	estimation network	k				
dk (14)	t_emb	32	32	8	4	4	4	8	32
# of Mamba layers	diff_e_layers	1	1	1	1	1	1	1	1
$H \text{ in } \mathcal{M}_{res,\theta}$ (14)	diff_d_model	512	512	$256^{\leq 48}, 512^{\geq 96}$	128	128	128	128	512
FCN dim in Mamba layers	diff_d_ff	2048	128	$256^{\leq 48}, 512^{\geq 96}$	128	128	128	128	512
dropout	diff_dropout	0.3	0.5	$0.5^{\leq 48}, 0.4^{\geq 96}$	0.7	0.7	0.7	0.5	$0.2^{\leq 96}$, $0.5^{\geq 192}$
learning rate	diff_learning_rate	0.0005	0.0005	0.0005	0.0003	0.0003	0.0003	$0.0005^{\leq 336}, 0.3^{720}$	$0.0001^{\leq 96}, 0.0005^{\geq 19}$
λ weight decay for Adam	weight_decay	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}
			Erro	r-aware expansion					
α (12	alpha					1			
			Cov	erage optimization					
γ (Section 3.3.2)	I	1			{(0.04, 0.08,	1}		

C Metrics

The mean absolute error (MAE) is a common metric used to measure the average magnitude of errors between predicted values and actual values in a dataset, without considering their direction (positive or negative). It is the L₁ difference between the predicted and true values given by

$$MAE(\mathbf{y}, \hat{\mathbf{y}}) = \|\mathbf{y} - \hat{\mathbf{y}}\|_{1}, \tag{19}$$

where n is the total number of data points, $\mathbf{y} = \{y_1, \dots, y_n\}$ represents the true value and $\hat{\mathbf{y}} = \{\hat{y}_1, \dots, \hat{y}_n\}$ represents the predicted value.

The mean squared error (MSE) is a commonly used metric in regression tasks. It measures the average squared difference between the predicted values and the true values given by

$$MSE(\mathbf{y}, \hat{\mathbf{y}}) = \|\mathbf{y} - \hat{\mathbf{y}}\|_{2}^{2}.$$
 (20)

It is well known that the MSE is sensitive to outliers.

The Continuous Ranked Probability Score (CRPS) is a statistical measure used to evaluate the accuracy of probabilistic forecasts for continuous variables by comparing the cumulative distribution function (CDF) of the forecast and the actual outcome. The CRPS is defined as

$$CRPS(F_{\hat{y}}, y) = \int_{-\infty}^{\infty} (F_{\hat{y}}(x) - \mathbb{1}(x \ge y))^2 dx,$$
(21)

where $F_{\hat{y}}(\cdot)$ is the forecasted CDF for the variable and $x \in \mathbb{R}$ is the observed value. The CRPS is different from MAE and MSE because it considers the whole distribution of the forecasts instead of point estimates.

The Predictive Interval Coverage Probability (PICP) quantifies the fraction of observed values that fall within a model's predicted intervals over a dataset. It is computed by averaging binary indicators that signal whether each actual outcome lies inside its associated predictive interval and comparing this empirical rate to the nominal confidence level (e.g., 90 %).

$$PICP(\gamma, \mathbf{y}, \mathbf{F}_{\hat{\mathbf{y}}}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1} \left\{ y_i \in \left[F_{\hat{y}, i}^{-1} \left(\frac{1-\gamma}{2} \right), F_{\hat{y}, i}^{-1} \left(\frac{1+\gamma}{2} \right) \right] \right\}, \tag{22}$$

where γ is the desired nominal coverage, n is the total number of data points, $\mathbf{y} = \{y_1, \dots, y_n\}$ represents the true value, $\mathbf{F}_{\hat{\mathbf{y}}} = \{F_{\hat{y},1}, \dots, F_{\hat{y},n}\}$ represents the CDF of the forecasts, and $F_{\hat{y},i}^{-1}$ is the inverse CDF (quantile function) of the predictive distribution for the i-th point. A PICP equal to the nominal level indicates perfect calibration, whereas values above or below it denote over-coverage or under-coverage, respectively.

We define the PICP distance metric as the sum of the absolute difference between the PICP and its nominal level for $\gamma \in \{0.5, 0.8, 0.95\}$:

PICP dis =
$$\sum_{\gamma \in \{0.5, 0.8, 0.95\}} |PICP(\gamma, \mathbf{y}, \mathbf{F}_{\hat{\mathbf{y}}}) - \gamma|.$$
(23)

This accounts for multiple prediction intervals simultaneously, making the metric more robust.

D Choice of MAE and MSE

When evaluating a predictive distribution using point metrics, it is more appropriate to first take the mean of the samples and then compute the metric, rather than computing the metric directly across all samples of the empirical distribution. Assuming $\mathbf{y} \in \mathbb{R}^n$ is the ground truth and $\hat{\mathbf{y}} = \{\hat{\mathbf{y}}_1, \dots, \hat{\mathbf{y}}_S\}$ are S predictions for \mathbf{y} , where $\hat{\mathbf{y}} \in \mathbb{R}^{S \times n}$, by the definition of the MAE in (19), let $\hat{\mathbf{y}} = \frac{1}{S} \sum_{i=1}^{S} \hat{\mathbf{y}}_i$, by applying the triangle inequality, we have

$$\mathrm{MAE}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{S} \sum_{i=1}^{S} \|\hat{\mathbf{y}}_i - \mathbf{y}\|_1 \ge \|\bar{\hat{\mathbf{y}}} - \mathbf{y}\|_1 = \mathrm{MAE}\left(\bar{\hat{\mathbf{y}}}, \mathbf{y}\right).$$

The above relation suggests that the MAE computed between the average of the S predictions and the ground truth is tighter than the average MAE computed between each individual prediction and the ground truth. Therefore, we adopt the former approach for evaluating MAE in this paper. Following a similar rationale, we evaluate MSE using the formula $MSE(\bar{\hat{y}}, y) = \|\bar{\hat{y}} - y\|^2$.

E CRPS-optimizing Standard Deviation

Assuming the ground truth is y and that the predictions \hat{y} follow a zero-mean normal distribution $\hat{y} \sim \mathcal{N}(0, \sigma^2)$. Then there exists an optimal σ , denoted by σ^* , that minimizes the CRPS such that

$$\sigma^* = \operatorname*{arg\,min}_{\sigma>0} \mathsf{CRPS}(F_{\hat{y},\sigma}, y),$$

where $F_{\hat{y},\sigma}$ is the CDF of the forecasted \hat{y} , parameterized by σ , and

$$CRPS(F_{\hat{y},\sigma}, y) = \int_{-\infty}^{\infty} (\Phi(x/\sigma) - \mathbb{1}_{x \ge y})^2 dx = \int_{-\infty}^{y} \Phi^2(x/\sigma) dx + \int_{y}^{\infty} (1 - \Phi(x/\sigma))^2 dx$$
$$= \int_{-\infty}^{y} \Phi^2(x/\sigma) dx + \int_{-\infty}^{-y} \Phi^2(x/\sigma) dx = \int_{-\infty}^{y} (1 + \mathbb{1}_{x \le -y}) \Phi^2(x/\sigma) dx,$$

where $\Phi(x)$ is the CDF of the standard normal distribution (recall that $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-u^2/2} du$). We obtain σ^* by solving

$$\frac{\partial \operatorname{CRPS}(F_{\hat{y},\sigma}, y)}{\partial \sigma} = 0, \tag{24}$$

where

$$\begin{split} &\frac{\partial \text{CRPS}(F_{\hat{y},\sigma},y)}{\partial \sigma} \\ &= -\frac{\sqrt{2}}{\sqrt{\pi}\sigma} \int_{-\infty}^{y} (1+\mathbbm{1}_{x\leq -y}) \Phi(x/\sigma) \frac{x}{\sigma} e^{-x^2/(2\sigma^2)} dx \\ &= -\sqrt{\frac{2}{\pi}} \int_{-\infty}^{y/\sigma} (1+\mathbbm{1}_{\tilde{x}\leq -y/\sigma}) \Phi(\tilde{x}) \tilde{x} e^{-\tilde{x}^2/2} d\tilde{x} \\ &= \sqrt{\frac{2}{\pi}} \left(\Phi(\tilde{x}) e^{-\tilde{x}^2/2} \Big|_{-\infty}^{y/\sigma} + \Phi(\tilde{x}) e^{-\tilde{x}^2/2} \Big|_{-\infty}^{-y/\sigma} \right) - \frac{1}{\pi} \int_{-\infty}^{y/\sigma} (1+\mathbbm{1}_{\tilde{x}\leq -y/\sigma}) e^{-\tilde{x}^2} d\tilde{x}, \end{split}$$

where the change of variables $\tilde{x} = x/\sigma$ and integration by parts are applied. The above derivation combining with (24) implies

$$\begin{split} \sqrt{\frac{2}{\pi}} \left(\Phi(y/\sigma^\star) + \Phi(-y/\sigma^\star) \right) e^{-y^2/(2\sigma^{\star\,2})} &= \frac{1}{\pi} \int_{-\infty}^{y/\sigma^\star} (1 + \mathbbm{1}_{\tilde{x} \le -y/\sigma^\star}) e^{-x^2} dx \\ &= \frac{1}{\sqrt{\pi}} \int_{-\infty}^{y/(\sqrt{2}\sigma^\star)} (1 + \mathbbm{1}_{x' \le -y/(\sqrt{2}\sigma^\star)}) e^{-x'^2/2} dx' \\ &= \frac{1}{\sqrt{\pi}} \left(\Phi\left(\frac{y}{\sqrt{2}\sigma^\star}\right) + \Phi\left(-\frac{y}{\sqrt{2}\sigma^\star}\right) \right), \end{split}$$

where the change of variables $x'=\tilde{x}\sqrt{2}$ is applied. By using the fact that $\Phi(a)+\Phi(-a)=1$ for any $a\in\mathbb{R}$, the last display implies $\sqrt{2}e^{-y^2/(2\sigma^{\star^2})}=1$, which further implies

$$\sigma^* = \frac{|y|}{\sqrt{\ln 2}}.$$

The derivation for the case when $\hat{y} \sim \mathcal{N}(\mu, \sigma^2)$ is similar. For brevity, we omit the details, and the optimal value of σ , denoted by σ^* , is given by

$$\sigma^* = \frac{|y - \mu|}{\sqrt{\ln 2}} = \frac{|\varepsilon|}{\sqrt{\ln 2}}.$$
 (25)

The above result shows that under the normal assumption of the predictions, if we know the absolute difference (absolute error ε) between the ground truth (y) and the mean of the predictions (μ) , we can minimize the CRPS by either *stretching* or *shrinking* the distribution of predictions so that the standard deviation of the distribution matches with the optimal standard deviation.

We verify the theory in Figure 2. In the figure, we plot the normalized CRPS (CRPS/ $|\varepsilon|$) against the normalized standard deviation of the predictions ($\sigma/|\varepsilon|$). According to (25), $\sigma^*/|\varepsilon| = 1/\sqrt{\ln 2} \approx 1.2011$, which corresponds to the global minimum indicated by the red line. We also observe in Figure 2 that the CRPS as a function of σ is convex: it decreases as $\sigma/|\varepsilon|$ approaches σ^* , and then increases rapidly as σ departs from σ^* . This phenomenon aligns with the experimental results reported in Appendix J.

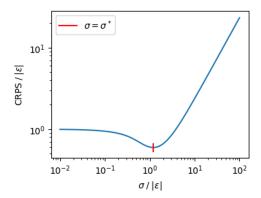


Figure 2: $\operatorname{CRPS}/|\varepsilon|$ vs $\sigma/|\varepsilon|$ when the predictions z are drawn from a normal distribution with mean μ and standard deviation σ , and the ground truth being y.

Let $F_{\hat{y}}(\cdot)$ be the CDF of the predictions for the ground truth y. If we condense the predictions to their mean (\bar{y}) , making the standard deviation of the distribution 0, then $F_{\hat{y}}(x) = \mathbb{1}\{x \geq \bar{\hat{y}}\}$, and by the definition of CRPS in (21),

$$\mathrm{CRPS}(F_{\hat{y}},y) = \int_{-\infty}^{+\infty} (F_{\hat{y}}(x) - \mathbbm{1}_{x \geq y})^2 \, dx = \int_{-\infty}^{+\infty} (\mathbbm{1}_{x \geq \bar{y}} - \mathbbm{1}_{x \geq y})^2 \, dx = |\bar{\hat{y}} - y| = \mathrm{MAE}(\bar{\hat{y}},y).$$

This relation shows that CRPS equals MAE when the predictive distribution has zero variance (i.e., is concentrated at the mean). For a Gaussian distribution, if the CRPS is larger than the MAE, then based on Figure 2, the standard deviation of the distribution σ must be greater than σ^* . This relation can be generalized for a multivariate y and $F_{\hat{\mathbf{v}}}$.

F Performance Promotion with Different Point Estimators

We use SMamba and TimeFilter as the point estimators, and present results comparing CRPS and MAE before and after applying RDIT (Table 6). Across six datasets and averaged over three different prediction lengths, we observe that regardless of the point estimator used, applying RDIT consistently improves the CRPS and often improves the MAE. The reason why point-based metrics such as MAE may not improve with RDIT is straightforward: since RDIT focuses on uncertainty modeling, the mean or median of the predictions is likely to remain unchanged. A similar effect on point-based metrics is also observed in Table 1 of the D3U literature [25], where equipping a point estimator with uncertainty modeling does not necessarily lead to improved point accuracy.

Table 6: CRPS and MAE results comparing SMamba and TimeFilter point estimators equipped with a zero-mean Gaussian and RDIT. Results are averaged over prediction lengths 24, 48, and 96.

Point Model		SMa	amba			TimeI	Filter	
Metric	CRPS	S	MAE	Ξ	CRPS	S	MAE	
Dataset	$\mid \mathcal{N}(0, \sigma_{trn}^2)$	+RDIT	$\mathcal{N}(0, \sigma_{trn}^2)$	+RDIT	$\mathcal{N}(0, \sigma_{trn}^2)$	+RDIT	$\mathcal{N}(0, \sigma_{trn}^2)$	+RDIT
Traffic	0.233	0.204	0.256	0.242	0.207	0.200	0.228	0.224
Weather	0.184	0.146	0.174	0.179	0.171	0.132	0.160	0.159
Electricity	0.172	0.169	0.305	0.220	0.162	0.161	0.209	0.210
Exchange	0.126	0.124	0.153	0.157	0.112	0.111	0.151	0.150
ETTm2	0.183	0.182	0.231	0.227	0.178	0.174	0.225	0.224
Solar	0.207	0.170	0.230	0.210	0.186	0.178	0.204	0.203

G Detailed Results for Probabilistic Metrics

Table 7 shows the CRPS across all prediction lengths, comparing RDIT with the baselines. Table 8 presents the corresponding PICP distance results. Together, these tables summarize CRPS and PICP distance performance over a wide range of prediction lengths. From the CRPS results, we observe that RDIT achieves the best performance on average, but the point-based models augmented with zero-mean Gaussians perform comparably—often outperforming recent state-of-the-art PTSF methods, except on the Solar dataset. In terms of PICP distance, these Gaussian-augmented point-based models generally underperform compared to some recent PTSF methods such as D3U. However, TimeDiff stands out among the point-based models, especially on the Electricity dataset. Although TimeDiff is fundamentally a TSF model, its use of diffusion may result in similar training error distributions between training and test datasets, contributing to its strong performance.

Table 7: CRPS of different algorithms for eight datasets across prediction lengths 24, 48, 96, 192, 336, and 720. **Bold**: best (lowest) value; underlined: second to best.

	o, and	_ ~ .		•••		(20.	. • 5	,	,					• • • • •			••								
	Dataset			Tra	ıffic					Wea	ather					Elect	ricity					Exch	ange		
Type	Horizon	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720
LLM	Chronos	0.311	0.362	0.401	N/A	N/A	N/A	0.104	0.155	0.208	N/A	N/A	N/A	0.226	0.254	0.283	N/A	N/A	N/A	0.082	0.115	0.174	N/A	N/A	N/A
	TimeDiff																			0.096					
	iTransformer																								
TSF	SMamba																			0.091					
	PatchTST																			0.082					
	TimeFilter	0.197	0.207	0.217	0.225	0.233	0.245	0.145	0.170	0.199	0.231	0.259	0.296	0.150	0.164	0.173	0.185	0.192	0.208	0.077	0.108	0.152	0.221	0.305	0.491
	SSSD	0.459	0.535	0.556	0.660	0.709	0.789	0.313	0.386	0.373	0.446	0.433	0.450	0.487	0.582	0.576	0.635	0.614	0.753	0.626	1.150	1.228	1.141	1.195	1.184
	Tactis-2	0.316	0.370	0.643	N/A	N/A	N/A	0.126	0.169	0.208	N/A	N/A	N/A	0.283	0.306	0.416	N/A	N/A	N/A	0.081	0.126	0.183	N/A	N/A	N/A
PTSF	TMDM	0.237	0.231	0.244	0.246	0.263	0.248	0.136	0.160	0.185	0.234	0.261	0.301	0.203	0.206	0.223	0.245	0.267	0.301	0.172	0.214	0.311	0.310	0.408	0.586
	D3U	0.206	0.213	0.221	0.228	0.244	0.259	0.124	0.140	0.182	0.204	0.242	0.366	0.167	0.189	0.196	0.198	0.213	0.249	0.088	0.116	0.184	0.260	0.397	0.513
	RDIT	0.184	0.207	0.209	0.217	0.226	0.241	0.108	0.129	0.158	0.204	0.242	0.273	0.149	0.163	0.170	0.184	0.197	0.207	0.075	0.107	0.151	0.218	0.305	0.484
				ET	Γm1					ET	Γm2					ET	Th1					Sol	ar		
LLM	Chronos	0.337	0.428	0.464	N/A	N/A	N/A	0.171	0.211	0.252	N/A	N/A	N/A	0.304	0.334	0.388	N/A	N/A	N/A	0.279	0.493	0.648	N/A	N/A	N/A
	TimeDiff	0.286	0.341	0.344	0.365	0.399	0.413	0.165	0.198	0.229	0.267	0.302	0.344	0.279	0.297	0.319	0.345	0.370	0.375	0.211	0.274	0.332	0.371	0.406	0.408
																				0.173					
TSF	SMamba																			0.174					
	PatchTST																			0.141					
	TimeFilter	0.216	0.250	0.271	0.293	0.310	0.340	0.153	0.177	0.203	0.241	0.274	0.323	0.271	0.289	0.307	0.328	0.345	0.358	0.147	0.194	0.217	0.237	0.245	0.254
	SSSD	0.518	0.606	0.608	0.626	0.637	0.675	0.733	1.045	1.254	1.273	1.193	1.271	0.487	0.614	0.769	0.724	0.711	0.720	0.461	0.402	0.445	0.454	0.520	0.542
	Tactis-2	0.251	0.295	0.291	N/A	N/A	N/A	0.170	0.298	0.409	N/A	N/A	N/A	0.305	0.549	0.669	N/A	N/A	N/A	0.184	0.211	0.253	N/A	N/A	N/A
PTSF	TMDM	0.280	0.343	0.366	0.383	0.422	0.445	0.177	0.230	0.251	0.335	0.474	0.565	0.351	0.364	0.389	0.421	0.500	0.516	0.166	0.220	0.217	0.213	0.250	0.231
	D3U	0.221	0.249	0.274	0.295	0.305	0.336	0.160	0.186	0.233	0.248	0.280	0.326	0.307	0.344	0.366	0.449	0.498	0.455	0.118	0.170	0.160	0.185	0.189	0.198
	RDIT	0.210	0.239	0.269	0.289	0.306	0.339	0.151	0.174	0.197	0.240	0.267	0.327	0.271	0.282	0.309	0.330	0.341	0.352	0.130	0.187	0.217	0.203	0.236	0.215

Table 8: PICP distance of different algorithms for eight datasets across prediction lengths 24, 48, 96, 192, 336, and 720. **Bold**: best (lowest) value; underlined: second to best.

	, ,						,		/		′ —			_											
	Dataset			Tra	ffic					Wea	ther					Elect	ricity					Exch	ange		
Type	Horizon	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720
LLM	Chronos	0.828	0.853	1.106	N/A	N/A	N/A	0.771	0.832	1.157	N/A	N/A	N/A	0.982	0.990	1.205	N/A	N/A	N/A	0.516	0.558	0.964	N/A	N/A	N/A
	TimeDiff																			0.212					
																				0.158					
TSF	SMamba																			0.147					
	PatchTST	0.382	0.374	0.374	0.373	0.376	0.355	0.481	0.460	0.433	0.408	0.385	0.393	0.202	0.200	0.198	0.195	0.189	0.175	0.152	0.142	0.132	0.109	0.064	0.271
	TimeFilter	0.375	0.370	0.365	0.359	0.352	0.345	0.440	0.416	0.386	0.357	0.343	0.302	0.170	0.155	0.152	0.134	0.122	0.106	0.187	0.149	0.146	0.138	0.088	0.115
	SSSD				1.815		1.901													1.537			1.789		
	Tactis-2	0.692	0.450	1.197	N/A		N/A					N/A			0.185					0.179			N/A		N/A
PTSF	TMDM	0.154	0.179	0.184	0.154	0.241	0.160	0.567	0.438	0.453	0.510	0.453	0.262	0.158	0.123	0.148	0.210	0.123	0.243	0.083	0.128	0.399	0.415	0.609	0.466
	D3U																			0.402					
	RDIT	0.216	0.088	0.136	0.151	0.123	0.264	0.103	0.229	0.141	0.201	0.097	0.118	0.054	0.048	0.092	0.115	0.056	0.105	0.035	0.026	0.128	0.032	0.015	0.116
		1		ET.	Γm1			1		ET.	Γm2					ET	Th1					Sol	ar		
LLM	Chronos	0.693	0.744	1.017	N/A	N/A	N/A	0.845	0.829	1.088	N/A	N/A	N/A	0.710	0.706	0.989	N/A	N/A	N/A	0.839	0.883	1.212	N/A	N/A	N/A
	TimeDiff	0.146	0.116	0.161	0.167	0.159	0.170	0.133	0.149	0.193	0.216	0.226	0.243	0.206	0.223	0.225	0.219	0.214	0.238	0.240	0.219	0.169	0.213	0.127	0.115
	iTransformer	0.139	0.134	0.151	0.192	0.204	0.205	0.150	0.168	0.220	0.250	0.265	0.272	0.156	0.188	0.192	0.190	0.198	0.197	0.331	0.290	0.291	0.279	0.267	0.269
TSF	SMamba	0.134	0.133	0.147	0.209	0.210	0.195	0.142	0.167	0.238	0.265	0.270	0.266	0.187	0.202	0.203	0.194	0.193	0.211	0.347	0.301	0.306	0.291	0.271	0.279
	PatchTST	0.145	0.171	0.171	0.182	0.197	0.206	0.140	0.189	0.226	0.257	0.271	0.268	0.171	0.194	0.203	0.219	0.222	0.225	0.384	0.328	0.313	0.286	0.285	0.294
	TimeFilter	0.167	0.165	0.170	0.193	0.200	0.215	0.163	0.181	0.223	0.252	0.265	0.273	0.184	0.204	0.219	0.220	0.230	0.238	0.378	0.327	0.291	0.278	0.271	0.269
	SSSD	1.097	1.216	1.087	1.078	1.171	1.184	1.508	1.781	1.802	1.708	1.537	1.655	0.787	0.918	1.068	0.885	1.090	1.260	1.673	1.491	1.260	1.551	1.478	1.279
	Tactis-2	0.468	0.154	0.164	N/A	N/A	N/A	0.407	0.694	0.941	N/A	N/A	N/A	0.297	0.924	1.324	N/A	N/A	N/A	0.796	1.377	1.642	N/A	N/A	N/A
PTSF	TMDM	0.233	0.257	0.312	0.429	0.470	0.458	0.625	0.597	0.299	0.531	0.695	0.823	0.306	0.376	0.327	0.412	0.629	0.513	0.327	0.360	0.192	0.278	0.276	0.288
	D3U	0.171	0.106	0.224	0.083	0.056	0.066	0.172	0.153	0.110	0.261	0.152	0.328	0.782	0.563	0.224	0.146	0.396	0.311	0.064	0.197	0.125	0.243	0.131	0.130
	RDIT	0.027	0.031	0.043	0.050	0.036	0.066	0.091	0.092	0.101	0.064	0.047	0.124	0.141	0.169	0.062	0.085	0.094	0.108	0.050	0.018	0.073	0.040	0.066	0.228

H Visualizations of Prediction Intervals

In this section, we visualize the prediction intervals on two datasets to showcase the effectiveness of RDIT for PTSF. Figure 3 visualizes the prediction intervals for the ETTh1 dataset. Although RDIT fails to provide an accurate predictive distribution around time points $t = \{210, 290, 310\}$, it overall yields better mean estimations and tighter intervals that capture the ground truth, resulting in lower CRPS and PICP distance. Similarly, Figure 4 shows the prediction intervals for the Exchange dataset. Since the Exchange dataset resembles a random walk, the uncertainty of predictions for all models increases further into the future. Nevertheless, RDIT provides a more stable mean prediction that remains close to the ground truth. Moreover, we observe that SMamba and TimeDiff produce biased estimations starting from the initial prediction point.

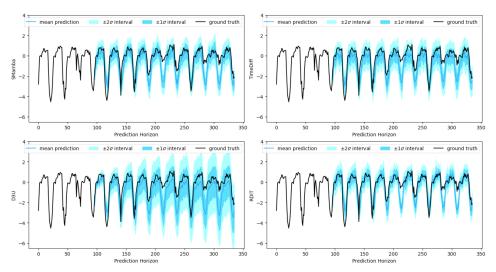


Figure 3: Visualization of prediction intervals for the ETTh1 dataset (variate #2) using SMamba, TimeDiff, D3U, and RDIT. For each time point t, the mean μ_t and standard deviation σ_t are calculated, and two intervals— $[\mu_t - \sigma_t, \mu_t + \sigma_t]$ and $[\mu_t - 2\sigma_t, \mu_t + 2\sigma_t]$ —are plotted over the prediction horizon. The leftmost section of each subfigure, which shows only the ground truth without intervals, represents the input history data provided to the prediction model.

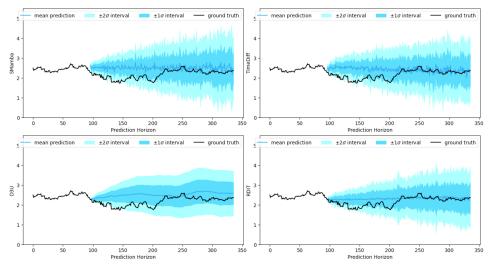


Figure 4: Visualization of prediction intervals for the exchange dataset (variate #7) using SMamba, TimeDiff, D3U, and RDIT. For each time point t, the mean μ_t and standard deviation σ_t are calculated, and two intervals— $[\mu_t - \sigma_t, \mu_t + \sigma_t]$ and $[\mu_t - 2\sigma_t, \mu_t + 2\sigma_t]$ —are plotted over the prediction horizon. The leftmost section of each subfigure, which shows only the ground truth without intervals, represents the input history data provided to the prediction model.

I Visualizations of Coverage Optimization

To provide evidence on why Coverage Optimization (CO) works, we plot the prediction intervals before and after applying CO and compare them to the ground truth, as shown in Figure 5. Initially, the ground truths are under-covered; that is, the actual observations fall outside the nominal intervals more frequently than expected. After applying CO, the prediction intervals tend to align more closely with the target coverage percentage. For the Solar dataset, CO introduces spikes in the prediction intervals over time, suggesting that while CO improves coverage probability, relying on it alone may negatively impact CRPS. This indicates that further optimization of the prediction distribution may be necessary.

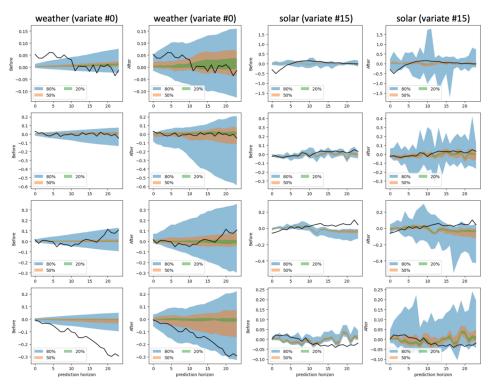


Figure 5: The ground truth and the 20%, 50%, and 80% prediction intervals plotted against the prediction horizon before and after coverage optimization (CO). The first and second columns correspond to the weather dataset before and after CO, respectively, while the third and fourth columns correspond to the solar dataset before and after CO. The plot scales between before and after CO are matched.

Figure 6 shows the PICP at different interval percentiles for the Weather and Solar datasets. By calibrating on the validation dataset and applying the expansion to the test dataset, we can shape the prediction intervals and effectively reduce the mismatch between the true and estimated residual distributions.

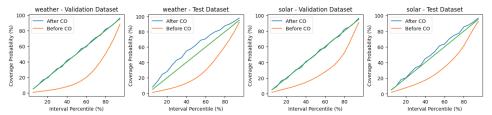


Figure 6: PICP plotted against interval percentile for the weather and solar dataset. The prediction length is 24. The validation datasets are used for calibrating and extracting target expansion factors to be applied to the test dataset.

J Error-aware Expansion

The EAE algorithm introduced in Section 3.3.1 requires access to the residuals in order to estimate the optimal standard deviation. In this section, we aim to answer the following question: Can we simply expand the distribution using a single fixed expansion factor (λ) to improve performance, potentially even outperforming EAE? Figure 7 shows the CRPS performance when applying a fixed λ for expansion, compared to using EAE. In this experiment, SMamba is used as $\mathcal{M}_{pt,\phi}$. Although simple expansion with λ as a hyperparameter may yield better performance in some cases, it requires searching over a wider range of values. Two main advantages of using EAE are evident: (1) the location of the minimum (arg min $_{\alpha}$ CRPS(\mathbf{F}_{α} , \mathbf{y})) falls within a relatively narrow range (between 1 and 5), and (2) applying EAE with $\alpha=1$ consistently improves performance compared to no expansion ($\lambda=1$). These observations highlight the stability and effectiveness of EAE.

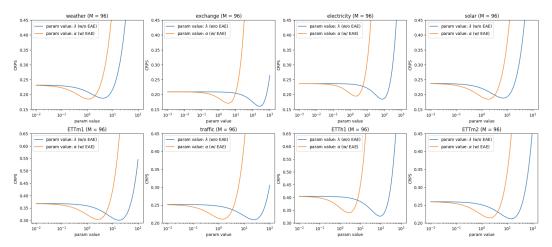


Figure 7: CRPS after applying simple expansion (blue) or EAE (orange) for a prediction length M=96. The x-axis (param value) corresponds to either λ (blue) or α (orange). The performance before any expansion is applied—i.e., directly using $\hat{\bf r}$ as the final residuals—corresponds to $\lambda=1$ on the blue line.

K Detailed Prediction Results for Point-based Metrics

Tables 9 and 10 show the MAE and MSE of all probabilistic forecasting models across all prediction lengths. In addition to the probabilistic metrics presented in the previous section, we observe that RDIT also excels in the point-based metrics MAE and MSE when compared with other PTSF methods across multiple prediction horizons. The ranks of the point metrics are more consistent, highlighting the stability of the point estimator, TimeFilter, used in the point prediction stage. Note that due to computational constraints, Chronos and Tactis-2 were not trained or evaluated on prediction lengths 192, 336, and 720.

Table 9: MAE of different algorithms for eight datasets across prediction lengths 24, 48, 96, 192, 336, and 720. **Bold**: best (lowest) value; underlined: second to best.

	Dataset			Tra	ıffic					Wea	ther		_	1		Elec	ricity					Excha	ange		
Type	Horizon	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720
LLM	Chronos	0.430	0.504	0.521	N/A	N/A	N/A	0.161	0.242	0.287	N/A	N/A	N/A	0.308	0.347	0.363	N/A	N/A	N/A	0.143	0.198	0.251	N/A	N/A	N/A
		0.00	0.586			0.777 N/A		0.396			0.545 N/A	0.520 N/A		0.575			0.729 N/A	0.716 N/A	0.849 N/A	0.729			1.267 N/A	1.325 N/A	
PTSF	TMDM	0.300	0.288	0.305	0.312	0.338	0.312	0.162	0.196	0.225	0.287	0.340	0.437	0.263	0.271	0.290	0.319	0.353	0.394	0.237	0.296	0.421	0.416	0.534	0.733
																				0.117					
				ET	Γm1					ET.	Γm2					ET	Th1					Sol	ar		
LLM	Chronos	0.500	0.638	0.647	N/A	N/A	N/A	0.273	0.343	0.369	N/A	N/A	N/A	0.442	0.492	0.525	N/A	N/A	N/A	0.410	0.696	0.827	N/A	N/A	N/A
			0.742			0.783 N/A	0.814 N/A	0.849		1.393	1.424 N/A	1.338 N/A		0.624			0.878 N/A	0.863 N/A	0.880 N/A	0.585	0.536		0.585 N/A	0.675 N/A	
PTSF	TMDM	0.357	0.437	0.465	0.489	0.539	0.581	0.218	0.281	0.317	0.423	0.588	0.746	0.448	0.468	0.503	0.544	0.652	0.687	0.200	0.279	0.272	0.268	0.317	0.292
	D3U RDIT	0.301 0.290	0.337 0.330	0.373 0.356	0.397 0.381									0.373 0.356						$\frac{0.176}{0.155}$					

Table 10: MSE of different algorithms for eight datasets across prediction lengths 24, 48, 96, 192, 336, and 720. **Bold**: best (lowest) value; underlined: second to best.

	,																								
	Dataset			Tra	ffic					Wea	ther					Elect	ricity					Excha	ange		
Type	Horizon	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720	24	48	96	192	336	720
LLM	Chronos	0.777	0.963	0.979	N/A	N/A	N/A	0.178	0.289	0.317	N/A	N/A	N/A	0.287	0.359	0.377	N/A	N/A	N/A	0.124	0.208	0.211	N/A	N/A	N/A
	SSSD	0.768	0.859	1.010	1.333	1.347												0.769			2.382	2.560	2.412	2.672	2.538
	Tactis-2	0.562	0.699	1.277	N/A	N/A	N/A	0.144	0.164	0.191	N/A	N/A	N/A	0.377	0.344	0.374	N/A	N/A	N/A	0.026	0.060	0.118	N/A	N/A	N/A
PTSF	TMDM	0.324	0.306	0.361	0.371	0.383	0.367	0.144	0.174	0.192	0.247	0.368	0.515	0.177	0.170	0.209	0.252	0.282	0.345	0.094	0.147	0.301	0.311	0.517	0.976
	D3U	0.454	0.510	0.523	0.527	0.549	0.784	0.107	0.131	0.197	0.224	0.294	0.478	0.126	0.160	0.167	0.175	0.198	0.258	0.029	0.049	0.109	0.225	0.484	0.767
	RDIT	0.285	0.305	0.333	0.350	0.368	0.398	0.088	0.114	0.146	0.192	0.238	0.314	0.096	0.115	0.126	0.144	0.143	0.178	0.022	0.044	0.086	0.181	0.345	0.849
				ET	Γm1					ET	Γm2					ET	Th1					Sol	ar		
LLM	Chronos	0.726	1.106	1.103	N/A	N/A	N/A	0.235	0.381	0.416	N/A	N/A	N/A	0.539	0.638	0.700	N/A	N/A	N/A	0.582	1.336	1.608	N/A	N/A	N/A
	SSSD	0.808	0.952	0.998	1.051	1.071	1.174	1.291	2.376	3.309	3.417	3.197	3.485	0.749	1.074	1.521	1.482	1.293	1.253	0.642	0.520	0.592	0.597	0.707	0.759
	Tactis-2	0.286	0.432	0.400	N/A	N/A	N/A	0.128	0.285	0.464	N/A	N/A	N/A	0.382	1.127	1.410	N/A	N/A	N/A	0.204	0.217	0.239	N/A	N/A	N/A
PTSF	TMDM	0.312	0.475	0.577	0.553	0.678	0.689	0.117	0.266	0.242	0.449	0.944	0.947	0.488	0.508	0.592	0.657	0.797	0.860	0.140	0.221	0.250	0.256	0.296	0.303
	D3U	0.221	0.276	0.325	0.382	0.399	0.483	0.105	0.139	0.213	0.252	0.314	0.415	0.329	0.394	0.445	0.620	0.612	0.658	0.113	0.187	0.190	0.229	0.244	0.269
	RDIT	0.223	0.278	0.312	0.356	0.390																		0.233	

L Change of CRPS in Denoising

Recall that $\hat{\mathbf{r}}^{\kappa_W} \sim \mathcal{N}(0,\mathbf{I})$ (Section 2). This means that if we directly take $\hat{\mathbf{r}}^{\kappa_W}$ and de-normalize it with σ_{trn} to match the standard deviation of the real residuals, then add it back to $\hat{\mathbf{y}}$, the result will be equivalent to equipping a point estimator with a zero-mean Gaussian—the strong baseline for reinforced TSF models. We show in Figure 8 that during the denoising process, the CRPS can improve *over* this baseline and yield a better final CRPS, with consistent improvements observed across all datasets. Despite these gains, for the Electricity, Exchange, ETTm1, and ETTh1 datasets, the CRPS improved by less than 1%, indicating that σ_{trn} already captures the uncertainty effectively compared to state-of-the-art PTSF methods. Moreover, in the Traffic, Electricity, Exchange, ETTm1, ETTh1, and Solar datasets, we observe that the CRPS of the final denoised step is worse than that of some intermediate steps. This suggests that the distribution of the residuals tends to overfit the training data, implying that the raw denoised results are suboptimal and further optimization, such as distribution matching, is necessary.

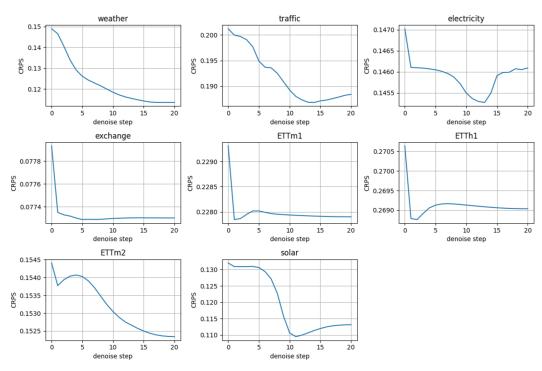


Figure 8: CRPS during each denoising step across eight datasets. The prediction length is 24. The starting point corresponds to a point estimator equipped with $\mathcal{N}(0, \sigma_{trn}^2)$. EAE and CO were not applied in these experiments.

M Visualization of the Effect of Error-aware Expansion

Error-aware Expansion (EAE) appropriately assigns a wider distribution to prediction regions with higher uncertainty. Indeed, an important aspect of PTSF is that the predictions should be uncertainty-aware. We provide visualizations as to how this works. Figure 9 shows the ground truth and predictions for the traffic dataset, along with the mean of the absolute value of the residuals ($\mathbb{E}[|\hat{\mathbf{r}}|]$). It can be seen that for regions where the original mean of the predictions ($\mathbb{E}[\hat{\mathbf{y}}+\hat{\mathbf{r}}]$) are farther away from the ground truth, the standard deviation after EAE are also relatively larger. This indicates that $\mathbb{E}[|\hat{\mathbf{r}}|]$ can indeed capture the uncertainty of the predictions and therefore optimize the CRPS.

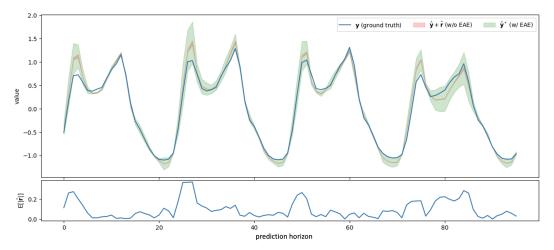


Figure 9: Visualization on the Traffic dataset (variate #800) with a prediction length of 96. SMamba is used as $\mathcal{M}_{pt,\phi}$, and the prediction mean is not confined to 0 in this experiment. Coverage Optimization (CO) is not applied. The shaded prediction area corresponds to the $\pm 1\sigma$ interval. The top plot shows the ground truth and predictions before and after applying EAE, while the bottom plot displays the corresponding mean of the absolute residuals ($\mathbb{E}[|\hat{\mathbf{r}}|]$).

N Limitations and Future Work

We point out several limitations and outline directions for future work. First, EAE currently works in theory only when predictions follow a Gaussian distribution; generalization to arbitrary predictive distributions is needed. Second, the PICP distance is computed using only three intervals; we aim to develop a more generalized metric that incorporates a broader range of intervals. Third, our results are not fully optimized with respect to hyperparameters; automatic hyperparameter tuning could further improve performance. Lastly, we assume that training errors follow a zero-mean Gaussian distribution, and future work can explore more flexible modeling of residuals.