SIN: Selective and Interpretable Normalization for Long-Term Time Series Forecasting

Lu Han [1 2] **Han-Jia Ye** [1 2] **De-Chuan Zhan** [1 2]

Abstract

In real-world applications, time series data frequently exhibit non-stationarity, with statistics changing over time. This variability undermines the forecasting accuracy of deep learning models that are trained on historical data but deployed for future prediction. A common approach to mitigate this issue involves normalizing the data to counteract statistical drift, followed by denormalization on the prediction. However, existing methods often employ heuristic normalization techniques that do not fully account for the unique characteristics of the series. Our paper addresses the critical question in this context: *which statistics should be removed and restored?* We argue that the statistics selected for normalization should exhibit both *local invariance and global variability* to ensure their *correctness and helpfulness*. To this end, we propose the Selective and Interpretable Normalization methodology, dubbed SIN. This approach maximizes the covariance between a given look-back window and its subsequent future values, thereby identifying key statistics for normalization and simultaneously learning the corresponding normalization transformations. The interpretable framework can be used to explain the success and limitations of some popular normalization methods. By integrating SIN, we demonstrate improvements in the performance of several prevalent forecasting models, thereby validating the utility of our approach.

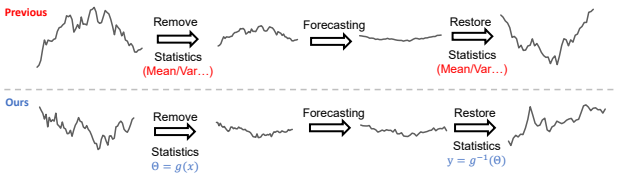


Figure 1. The normalization method removes the statistics from the input and restores the statistics to the model's prediction. This process mitigates statistics drift that happens globally. Previous methods usually use heuristic normalizations that ignore the unique statistics of each time series. In contrast, our method overcomes the limitations by using a learned normalization method.

1. Introduction

Time series forecasting plays a pivotal role in numerous domains, including energy (Kardakos et al., 2013), transportation (Kadiyala & Kumar, 2014), and healthcare (Morid et al., 2023). In recent years, deep learning has emerged as a dominant force in this field and brought about a significant shift from traditional statistical methods (Box et al., 2015) to various kinds of neural networks (Hochreiter & Schmidhuber, 1997; Cho et al., 2014; Rangapuram et al., 2018; Bai et al., 2018; Franceschi et al., 2019; Zhou et al., 2021; Wu et al., 2021a; Zhou et al., 2022; Liu et al., 2022b). These methods demonstrated improved performance over traditional statistical approaches, particularly in handling high-dimensional series with long-term dependencies.

However, the efficacy of deep learning models in time series forecasting is often hindered by the inherent non-stationarity in time series (Hyndman & Athanasopoulos, 2018; Petropoulos et al., 2022). Non-stationarity, characterized by time-varying statistical properties such as mean, and variance, presents a formidable challenge. Models trained on historical data may struggle with future series that exhibit different distributions (Kim et al., 2021; Han et al., 2024).

In response to this challenge, the normalization method has become a critical step, as depicted in Figure 1. By removing the local statistics from the series, normalization mitigates the effects of distribution drift between seen and unseen data, enhancing the model's ability to generalize from past observations to future deployment (Passalis et al., 2020;

[1]School of Artificial Intelligence, Nanjing University, China [2]National Key Laboratory for Novel Software Technology, Nanjing University, China. Correspondence to: Han-Jia Ye <yehj@lamda.nju.edu.cn>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

Kim et al., 2021; Fan et al., 2023). However, the effectiveness of normalization is contingent upon two key decisions: the selection of statistics to normalize and the method of normalization. Traditional approaches often resort to heuristic techniques such as shifting the series to zero mean (Kim et al., 2021), but these approaches may not fully capture the unique aspects and dynamics of the time series.

Recognizing the limitations of existing methods, this paper reevaluates the role of normalization and focuses on the critical question: *Which statistics should be removed and restored?* We pinpoint two important properties that the statistics should possess – local invariance and global variability. The local invariance property indicates that the statistics should remain unchanged or change slowly among the look-back history and the forecasting future so that the remove and restore operation is valid. The global variability says the statistics should have a large variance over time, which is challenging for the base model to learn, necessitating normalization to prevent overfitting on training data.

To this end, we propose Selective and Interpretable Normalization (SIN) to learn how to normalize. SIN utilizes Partial Least Squares (PLS), a statistical method known for its ability to handle high-dimensional data and uncover the latent structures that explain covariance between variables. By maximizing covariance between adjacent past and future, SIN effectively discerns critical statistics for normalization and devises optimal normalization strategies. This dual functionality ensures the selected statistics exhibit both local invariance and global variability, while also providing interpretable normalization transformations. Our empirical analysis demonstrates that, while the mean is often the most explanatory statistic for covariance in many time series, it is not universally optimal, as evidenced by the significant role of the sinusoidal wave in explaining covariance. Our SIN method adeptly learns and applies these pertinent statistics for normalization. When integrated into various contemporary models, our SIN method demonstrates a substantial improvement in forecasting performance. Our contributions are outlined as follows:

1. We rethink the normalization process in time series and identify two properties that the statistics should possess – local invariance and global variability.
2. We propose Selective and Interpretable Normalization (SIN) to learn to select and perform normalization by partial least squares that maximize the covariance of statistics between history and the future. Experiments show its effectiveness in improving the performance of various recent forecasting models.
3. The SIN is interpretable and is a generalization of popular normalization methods like shifting to zero means. The success and limitations of these methods on different kinds of series can be explained in our framework.

2. Related Work

Time series forecasting. Time series forecasting is a critical area of research that finds applications in both industry and academia. With the powerful representation capability of neural networks, deep forecasting models have undergone a rapid development (Lim & Zohren, 2020; Wu et al., 2020; 2021b; Cirstea et al., 2018; Cui et al., 2021). Two widely used methods for time series forecasting are recurrent neural networks (RNNs) and convolutional neural networks (CNNs). RNNs model successive time points based on the Markov assumption (Hochreiter & Schmidhuber, 1997; Cho et al., 2014; Rangapuram et al., 2018), while CNNs extract variation information along the temporal dimension using techniques such as temporal convolutional networks (TCNs) (Bai et al., 2018; Franceschi et al., 2019). However, due to the Markov assumption in RNN and the local reception property in TCN, both of the two models are unable to capture the long-term dependencies in sequential data. Recently, the potential of Transformer models for long-term time series forecasting tasks has garnered attention due to their ability to extract long-term dependencies via the attention mechanism (Zhou et al., 2021; Wu et al., 2021a; Zhou et al., 2022). Nevertheless, Zeng et al. (2023) have highlighted that Transformers are less effective than a simple linear model. Further analysis by Han et al. (2024) attributes the phenomenon to the lack of robustness to resist the distribution drift at test time.

Non-stationarity of time series. Non-stationary time series are those where statistical properties, such as mean and variance, change over time. It is a critical aspect of statistical modeling and forecasting, particularly in fields such as economics, finance, and environmental science. Non-stationary time series often exhibit trends, seasonality, or varying volatility, making them more complex to analyze (Madsen, 2007). Traditional methods usually stationarize the time series to make them more predictable. For example, Box & Jenkins (1968) use the differencing method to make a non-stationary time series stationary by subtracting the previous observation. Decomposition methods model the trend and seasonality in the series and then remove them to get a stationary series (Cleveland et al., 1990; Dagum & Bianconcini, 2016; Wu et al., 2021a; Zhou et al., 2022). Even though these models may capture the non-stationarity within the seen series, they are vulnerable to the inherent distribution drift in unseen series (Kim et al., 2021; Han et al., 2024).

Normalization in time series forecasting. Unlike traditional drift challenges in machine learning, the target follows closely after the input in time series tasks, thus the target and input are highly correlated. Based on this, normalization techniques have become a focal point in recent research.

These methods strive to mitigate non-stationary elements and align data to a consistent distribution. DAIN (Passalis et al., 2020) introduces an innovative non-linear network. This network is adept at adaptively normalizing each input instance. ST-norm (Deng et al.) contributes by presenting dual normalization modules, focusing on both temporal and spatial dimensions of data. Subsequent research, however, highlights a critical insight: non-stationary factors are not mere noise but integral components for accurate forecasting. The removal of these elements can potentially lead to subpar predictions. Addressing this issue, RevIN (Kim et al., 2021) proposes a novel symmetric normalization method. It involves normalizing the input sequences and subsequently applying denormalization to the model's output sequences, leveraging instance normalization (Ulyanov et al., 2016). Building on this concept, Non-stationary Transformers (Liu et al., 2022c) introduce an innovative de-stationary attention mechanism within self-attention frameworks. This inclusion significantly enhances the performance of transformer-based models by integrating non-stationary factors. Recent advancement in this field is presented in Fan et al. (2023), where the study identifies both intra- and inter-space distribution shifts in time series data. The proposed solution focuses on learning distribution coefficients to address these shifts effectively. Lastly, SAN (Liu et al., 2023) extends the normalization concept further. It moves beyond the instance level, applying normalization at the slice level, thereby opening new avenues in handling non-stationary time series data.

However, these methods apply normalization in a heuristic way, usually by subtracting the mean and dividing the standard deviation. However, these heuristically selected statistics are not enough to fully describe the local invariance in time series data. Recognizing the limitations of previous methods, we propose a novel method to select the statistics and learn the normalization transformation.

3. Preliminaries

3.1. Time Series Forecasting

Time series forecasting deals with time series data that contain one or more variables, or channels, at each time step. Given historical values $\mathcal{X} \in \mathbb{R}^{L \times C}$, where L represents the length of the look-back window and C is the number of channels. The goal is to predict the future values $\mathcal{Y} \in \mathbb{R}^{H \times C}$, where $H > 0$ is the forecast horizon. The objective of the forecasting model is the minimize the forecasting risk $\mathcal{R}$:

$$\min_f \mathcal{R}(f) = \min_f \mathbb{E}_{t \in [T]} \ell(f(\mathcal{X}_t), \mathcal{Y}_t). \tag{1}$$

ℓ is the regression loss, which is usually the MSE loss (Zhou et al., 2021; Wu et al., 2021a; Zhou et al., 2022).

3.2. Normalization in Time Series

A major challenge in time series forecasting is the distribution drift caused by the non-stationarity of the data. Forecasting models, trained on historical data, may underperform when confronted with unseen data exhibiting a distinct distribution. Distinct from general machine learning scenarios, in time series forecasting, the target output is closely correlated with the input. Therefore, normalization methods are popular to be applied to solve the distribution drift by removing and subsequently transferring statistical properties from the historical input to future predictions. Formally, given the history input, the normalization method first computes the statistics Θ_t with a specific function g:

$$\Theta_t = g(\mathcal{X}_t). \tag{2}$$

Then a normalization function h is used to remove the statistics from the input:

$$\tilde{\mathcal{X}}_t = h(\mathcal{X}_t, \Theta_t) \tag{3}$$

The forecasting model predicts the normalized future values $\tilde{\mathcal{Y}}_t$ based on the normalized input $\tilde{\mathcal{X}}_t$:

$$\tilde{\mathcal{Y}}_t = f(\tilde{\mathcal{X}}_t) \tag{4}$$

Lastly, a denormalization function h^* is used to get the final prediction $\hat{\mathcal{Y}}_t$ based on the normalized values and the statistics:

$$\hat{\mathcal{Y}}_t = h^*(\tilde{\mathcal{Y}}_t, \Theta_t) \tag{5}$$

Traditional normalization methods primarily apply heuristics, such as calculating mean and variance, to compute these statistics. However, such heuristic methods may not fully leverage the distinctive characteristics of each time series dataset.

4. Methodology

The previous section describes the framework of normalization-based methods to alleviate the distribution drift of time series forecasting. In this section, we first rethink the role of normalization and argue about two properties that the transferred statistics should possess – local invariance and global variability. Then, based on the idea, we propose a simple method called Selective and Interpretable Normalization (SIN) to learn the normalization and denormalization pair by the partial least square method. Last, we explain how to forecast with the learned normalization.

4.1. Properties for Normalization

The key idea of normalization is to transfer the statistics of the input history to the future prediction. This poses unique requirements for the properties of the statistics. Here we identify two important properties that statistics should possess, named local invariance and global variability.

Definition 4.1 (Local Invariance)**.** Given a similarity function sim, the statistics computation function g, and a set

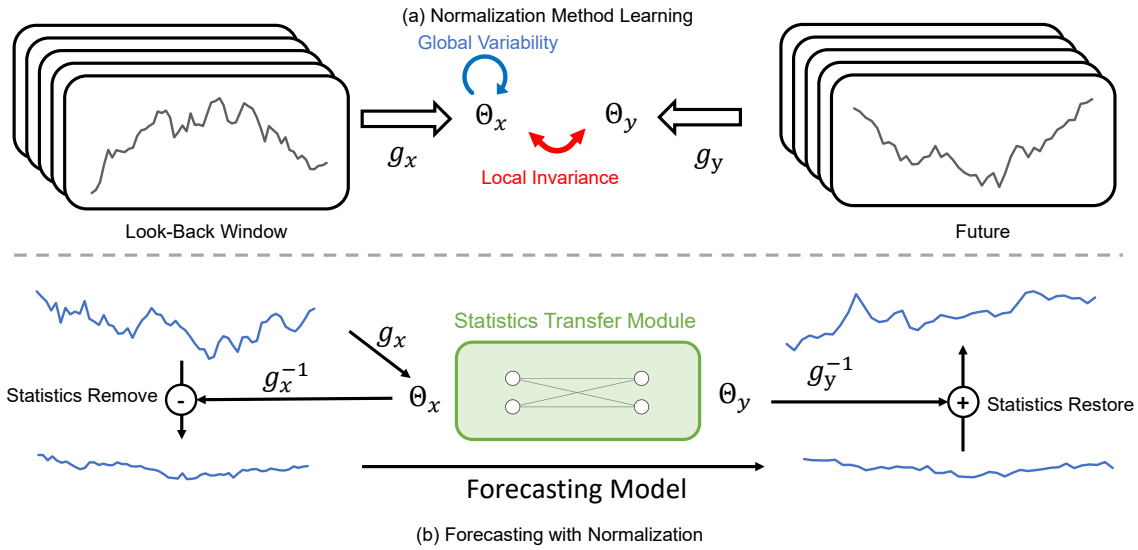


Figure 2. The illustration of the Selective and Interpretable Normalization (SIN) method. SIN first learns the normalization method by maximizing the local invariance and global variability of the statistics between the lookback window and the future. Then we use the learned normalization method to remove the statistics from the original series and restore them to the forecasting values. This process captures the invariance in times series and makes the forecasting model less vulnerable to distribution drift on unseen data. SIN uses a simple linear model that makes the normalization interpretable and helpful to reveal the characteristics of each time series.

of the look-back series x_t and the corresponding futures y_t, the local invariance of g is measured as $\sum_t sim(x_t, y_t)$

Keeping the local invariance large ensures the *correctness* of the normalization and the denormalization operation for time series forecasting. The statistics should remain unchanged within the prediction window to enable their accurate transference from historical data to future predictions. However, local invariance alone is insufficient for the full efficacy of normalization. For instance, a function $g(\cdot) = 0$ achieves perfect local invariance but contributes nothing to time series prediction accuracy. Hence, an additional property is required:

Definition 4.2 (Global Variability)**.** Given the statistics computation function g, and a set of the sub-series x_t, the global variability of g is measured as the variance of the statistics $Var[g(x_t)]$.

Keeping global variability large ensures the *helpfulness* of the normalization function in predicting future values according to the given history. These global varying statistics pose challenges for forecasting models to capture and put the model at risk of overfitting. Thus, a well-designed normalization method mitigates the risk of misinterpreting unseen data series removing these globally variable statistics.

4.2. Learning Normalization

Building on the previously outlined properties essential for effective normalization, we introduce a straightforward methodology to identify the optimal statistics for normalization. Given that the historical input and future values may differ in sequence lengths and necessitate distinct computational approaches, we define two statistical functions, g_x for the input and g_y for the future values. For simplicity, we assume that g_x is a linear projection parameterized by a unit vector $\boldsymbol{u} \in \mathbb{R}^L$. In other words, $g_x(\boldsymbol{x}) = \boldsymbol{u}^\top \boldsymbol{x}$. Similarly, $g_y(\boldsymbol{y}) = \boldsymbol{v}^\top \boldsymbol{y}, \boldsymbol{v} \in \mathbb{R}^H, \|\boldsymbol{v}\| = 1$. Firstly, we maximize the local invariance as shown in Definition 4.1. We select the negative Euclidean distance as the similarity measure. As a result, the local invariance loss takes the following form:

$$L_{\text{loc\_inv}} = \sum_t (g_x(\boldsymbol{x}_t) - g_y(\boldsymbol{y}_t))^2 \tag{6}$$

To maximize the variance on these statistics according to Definition 4.1, the global variability loss has the following form:

$$L_{\text{glo\_var}} = -\operatorname{Var}[g_x(\boldsymbol{x})] - \operatorname{Var}[g_y(\boldsymbol{y})] \tag{7}$$

It is safe to assume that the time series are centered along the series with zero mean without loss of generality. Consequently, adding the two losses together will result in a simplified objective:

$$\begin{aligned} \arg\min_{\boldsymbol{u},\boldsymbol{v}} \quad & L_{\text{loc\_inv}} + L_{\text{glo\_var}} \\ & = \|\boldsymbol{X}\boldsymbol{u} - \boldsymbol{Y}\boldsymbol{v}\|^2 - \|\boldsymbol{X}\boldsymbol{u}\|^2 - \|\boldsymbol{Y}\boldsymbol{v}\|^2 \\ & = -2\boldsymbol{u}^\top \boldsymbol{X}^\top \boldsymbol{Y}\boldsymbol{v} \\ \text{s.t.} \quad & \|\boldsymbol{v}\| = 1, \|\boldsymbol{u}\| = 1. \end{aligned} \tag{8}$$

Here, we define $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T)^\top$,$\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T)^\top$ as the set of the subseries.[1] Equation (8) is a framework that aims to maximize the covariance between the computed statistics of $\boldsymbol{X}$ and $\boldsymbol{Y}$. For simplicity and interpretability, we only use the linear projection here. However, the framework can be easily extended to the nonlinear scenarios by kernel methods or neural networks (Lindgren et al., 1993). Equation (8) also corresponds to the classical Partial Least Squares (PLS) problem that finds linear projection on the predicted variables and the observable variables to a new space. In the new space, the covariance between the two variables is explained maximally (Abdi, 2010). Many methods can be applied to solve the PLS problem (Dayal & MacGregor, 1997; Trygg & Wold, 2002). Here, we apply the PLS-SVD method that can efficiently compute the projection $\boldsymbol{u}$ and $\boldsymbol{v}$. Concretely, the Singular Value Decomposition (SVD) of the covariance matrix takes the following form:

$$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{X}^\top\boldsymbol{Y}.$$

where $\boldsymbol{U}$, $\boldsymbol{V}$ are orthogonal matrix. The diagonal matrix $\boldsymbol{\Sigma}$ measures the importance of each left and right singular vector pair. PLS-SVD algorithm just takes the top left/right singular vectors as the linear projection on $\boldsymbol{x}$ and $\boldsymbol{y}$.

In the experiments, we find that the singular values decay drastically to zero, and the projection with singular values larger than $\tau = 0.05 \times \max\{L, H\}$ usually exhibits a meaningful pattern and better results. Therefore, we select the singular vectors with singular values larger than τ as the projection functions. Denoting the selected singular vectors as $\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{V}}$, the learned normalization functions take the form:

$$g_x(\boldsymbol{x}) = \tilde{\boldsymbol{U}}^\top\boldsymbol{x}, \quad g_y(\boldsymbol{y}) = \tilde{\boldsymbol{V}}^\top\boldsymbol{y} \tag{9}$$

For each channel, we learn the normalization methods independently.

Discussion. It is easy to see that by setting $\boldsymbol{u} = \frac{1}{\sqrt{L}}\mathbf{1}_L, \boldsymbol{u} = \frac{1}{\sqrt{H}}\mathbf{1}_H$ where $\mathbf{1}_n$ is all 1 vector of dimension n, SIN is strictly equal to the mean shifting methods in instance normalization. SIN also include the discrete cosine transformation by setting $\boldsymbol{u} = \frac{\boldsymbol{c}}{\|\boldsymbol{c}\|}, \boldsymbol{c} = \{\cos(\frac{\pi k}{L} + a)\}_{k=1}^{L}$. In Section 5.2, we show these transformations contribute differently on different series. The comparison results of the mean shifting, discrete cosine transformation and SIN are shown in Appendix C.

4.3. Forecasting with Normalization

Normalization. Once we learn the normalization method g_x and g_y from the previous subsection, we apply the normalization method to the forecasting task to help the model improve the forecasting performance. To avoid confusion of symbols, we omit the channel symbol for each series since we apply normalization for each channel independently. Given a look-back window $\mathcal{X}$, denote the series for a single channel as $\boldsymbol{x} \in \mathbb{R}^L$. We first compute the statistics by g_x:

$$\Theta_x = g_x(\boldsymbol{x}) = \tilde{\boldsymbol{U}}^\top\boldsymbol{x}$$

Then we normalize the series by subtracting the reconstruction from the statistics:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} - g_x^{-1}(\Theta_x) = \Theta_x\tilde{\boldsymbol{U}}$$

Here the inverse function g_x^{-1} is computed as the linear projection to the original space, namely, $g_x^{-1}(\Theta_x) = \min_{\boldsymbol{U}^\star}\|\Theta_x\boldsymbol{U}^\star - \boldsymbol{x}\| = \Theta_x\tilde{\boldsymbol{U}}$. This normalization removes the local invariant and global variable statistics from the series, leaving the dynamics component that does not change much to the forecasting model. This process fully utilizes the capability of predicting the dynamics of time series, as well as reducing the risk of overfitting caused by high-variance features.

Denormalization. The learned statistics computation functions are locally invariant among the history-future pair. Thus, the statistics can be easily transferred from the past to the future. Following the normalization practice in (Kim et al., 2021), we use a simple affine model to transfer the statistics from $\boldsymbol{x}$ to $\boldsymbol{y}$. Concretely, the module is parameterized as ϕ and the statistics of $\boldsymbol{y}$ is computed as the following form:

$$\Theta_y = \phi \odot \Theta_x$$

where $\odot$ is the element-wise product. Then given the model prediction $\tilde{\boldsymbol{y}}$ we restore the statistics by the following equation:

$$\hat{\boldsymbol{y}} = \tilde{\boldsymbol{y}} + g_y^{-1}(\Theta_y) = \tilde{\boldsymbol{y}} + \sqrt{\frac{H}{L}}\Theta_y\tilde{\boldsymbol{V}}$$

Here, we align the scale of the inverse transformations g_x^{-1} and g_y^{-1} by $\sqrt{\frac{H}{L}}$.

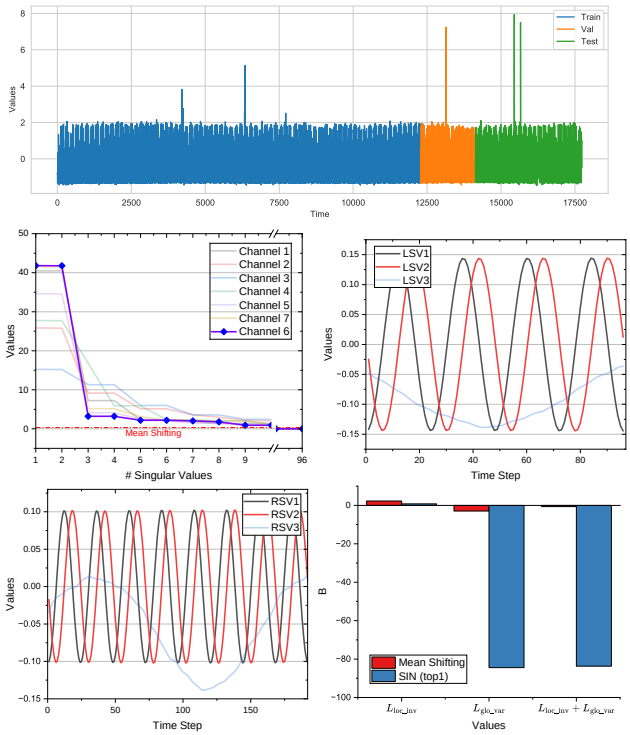
5. Analysis and Experiment

In this section, we conduct experiments to show the effectiveness of our method. We first give an analysis of the learned transformation and reveal the characteristics of each time series dataset. We then show that our method can improve forecasting models on various kinds of datasets.

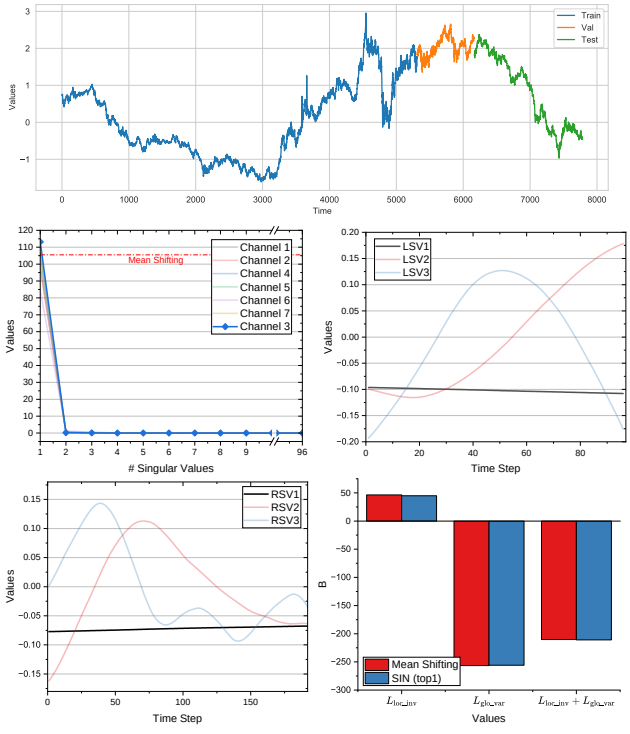
5.1. Setup

Datasets. We conduct extensive experiments on 11 widely used, real-world datasets that cover five mainstream time series forecasting applications, namely energy, electricity, traffic, economics, and weather. The datasets include: (1)

[1] Here, we omit the channel dimension since we compute the statistics loss for each channel independently.



(a) Traffic (Seasonal Series)



(b) Exchange-Rate (Trend Series)

Figure 3. Visualization of learned SIN transformation on two typical datasets. (1) The upper is a typical channel. (2) The mid-left is the singular values of each channel. (3) The mid-right and bottom-left figures show the left and right singular vectors (transformation vectors on the look-back window and the future). (4) Comparison between the learned SIN transformation and the commonly used mean shifting method with the local invariance loss and global variability loss.

ETT (Electricity Transformer Temperature) (Zhou et al., 2021) comprises two hourly-level datasets (ETTh) and two 15-minute-level datasets (ETTm). Each dataset contains seven oil and load features of electricity transformers from July 2016 to July 2018. (2) **Traffic** describes the road occupancy rates. It contains the hourly data recorded by the sensors of San Francisco freeways from 2015 to 2016. (3) **Electricity** collects the hourly electricity consumption of 321 clients from 2012 to 2014. (4) **Exchange-Rate** (Lai et al., 2018) collects the daily exchange rates of 8 countries from 1990 to 2016. (5) **Weather** includes 21 indicators of weather, such as air temperature, and humidity. Its data is recorded every 10 min for 2020 in Germany. (6) **ILI** describes the ratio of patients seen with influenza-like illness and the number of patients. It includes weekly data from the Centers for Disease Control and Prevention of the United States from 2002 to 2021. (7) **Solar**-Energy (Lai et al., 2018) records the solar power production of 137 PV plants in 2006, which is sampled every 10 minutes. (8)**PEMS** (Liu et al., 2022a) contains public traffic network data in California collected by 5-minute windows.

Forecasting models. SIN is a model-agnostic method that can be integrated with arbitrary forecasting models. To evidence the effectiveness of the method, we select some mainstream models based on different architectures and evaluate their performance for long-term multivariate time series forecasting: Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021a) and FEDformer (Zhou et al., 2022). We follow the implementation and settings provided in the official code of FEDformer [2] to implement these models.

Experiments details. The statistics transfer module is a simple affine model with parameter $\phi \in \mathbb{R}^m$, where m is the number of selected singular vectors. We use ADAM (Kingma & Ba, 2017) as the default optimizer across all the experiments and report the mean squared error (MSE) and mean absolute error (MAE) as the evaluation metrics. A lower MSE/MAE indicates a better performance. All the experiments are implemented by PyTorch (Paszke et al., 2019) and are conducted for three runs with a fixed random seed on a single NVIDIA RTX 3090 24GB GPU.

5.2. Interpretable Analysis

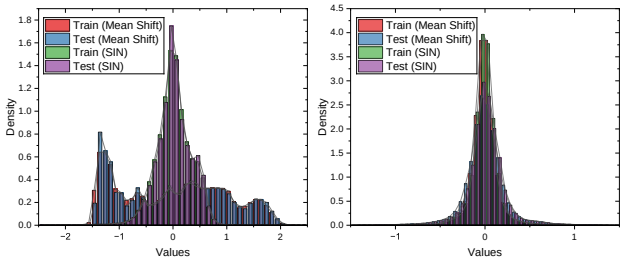
Our SIN uses a simple linear projection to model the normalization transformation, which makes our method interpretable to inspect the characteristics of each time series. In this subsection, visualize the learned transformations on different series and compare them under these visualizations.

Visualization of the transformations. Here, we experiment with the case that $L = 96, H = 192$. To understand

[2] https://github.com/MAZiqing/FEDformer

Table 1. Long-term multivariate forecasting errors with prediction lengths $H \in \{12, 24, 48, 96\}$ for PEMS datasets and $H \in \{96, 192, 336, 720\}$ for others. We fix the lookback length T = 96. All the results are *averaged from all prediction lengths*. Results of all prediction lengths are provided in Appendix F.

	Autoformer		+SIN		FEDformer		+SIN		Informer		+SIN	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	0.479	0.486	**0.470**	**0.478**	**0.455**	**0.432**	0.458	**0.432**	0.830	1.092	**0.518**	**0.555**
ETTm1	0.516	0.588	**0.434**	**0.440**	0.451	0.438	**0.409**	**0.392**	0.699	0.886	**0.453**	**0.466**
PEMS03	0.606	0.660	**0.346**	**0.241**	0.299	0.184	**0.293**	**0.170**	0.290	0.189	**0.239**	**0.136**
PEMS04	0.649	0.745	**0.341**	**0.230**	0.264	0.141	**0.250**	**0.130**	0.234	0.124	**0.224**	**0.115**
PEMS07	0.675	0.850	**0.353**	**0.258**	0.248	0.144	**0.241**	**0.135**	0.264	0.197	**0.219**	**0.140**
PEMS08	0.692	0.866	**0.475**	**0.499**	0.344	0.271	**0.335**	**0.261**	0.333	0.335	**0.290**	**0.233**
Electricity	0.327	0.214	**0.305**	**0.198**	0.322	0.209	**0.293**	**0.183**	0.414	0.329	**0.319**	**0.217**
Exchange	0.509	0.522	**0.492**	**0.445**	0.496	0.506	**0.494**	**0.448**	1.007	1.632	**0.472**	**0.462**
Solar	0.653	0.728	**0.398**	**0.315**	0.426	0.353	**0.343**	**0.263**	0.257	0.233	**0.252**	**0.226**
Traffic	0.379	0.615	**0.351**	**0.539**	0.373	0.605	**0.338**	**0.519**	0.444	0.769	**0.429**	**0.703**
Weather	0.361	0.320	**0.345**	**0.311**	0.456	0.453	**0.326**	**0.286**	0.557	0.634	**0.331**	**0.271**



(a) Traffic (b) Exchange

Figure 4. Distribution on train and test data after being normalized by mean shifting and SIN. Our SIN method can learn to transform the data into more normal distributions.

the learned transformations of SIN and the properties of each time series data, we selectively visualize the following things on two typical time series data: (1) upper is the plot of the univariant series on a selected channel. This channel is high on the mid-left figure with markers on its line. (2) mid-left is the singular value arranged in descending order. We highlight a typical example of these channels. The horizon line annotated as mean shifting is the case that $\boldsymbol{u}'$ and $\boldsymbol{v}'$ as the unit vector. The mean shifting line is located at $(\boldsymbol{u}')^\top \boldsymbol{X}^\top \boldsymbol{Y} \boldsymbol{v}'$, which proportion to the $L_{\text{loc\_inv}} + L_{\text{glo\_var}}$, (3) The mid-right and the bottom-left shows the first three left singular vectors and right singular vectors. The left/right singular vectors with corresponding singular values larger than $\tau = 0.05 * \max\{96, 192\}$ are highlighted. (4) the bottom right figure shows the value of local invariance, global variability, and the combination of the two losses with the mean shifting and the learned SIN transformation represented by the top-1 singular vectors pair. The results are displayed in the 3. From the figure, we got the following conclusions.

SIN adopts different normalization transformations for different datasets. Figure 3 presents two typical datasets – **Traffic** with mainly seasonal series and **Exchange-Rate** with mainly trended series. The spectrum (singular values) of the two datasets shows different patterns. For the seasonal series, the spectrum is usually dominated by two singular values. These two values correspond to two different harmonic waves with the same frequency but different phases. These waves represent the main periods for the series. For example, the traffic data is sampled hourly, and therefore the main period of traffic is 24, in accord with the period learned by SIN. For the trended series, the spectrum is dominated by the first singular value ($> 99\%$ for all channels). The corresponding singular vector is approximately the constant vector, equivalent to the mean shifting operation. The SIN can automatically adopt different strategies for different time series.

Success and limitation of mean shifting. Mean shifting is an important operation in instance normalization that has been a necessary preprocess for many state-of-the-art methods. This transformation is a special case in our framework by setting $\boldsymbol{u}$ and $\boldsymbol{v}$ to constant vectors. From our visualization, we can compare this operation to our SIN framework. On the trended dataset like Exchange-Rate, the mean shifting operation is especially useful because it achieves nearly the largest singular value for a dataset of this kind. It is also close to the transformation learned by SIN. The mean of the series achieves both low local invariance loss and global variability loss. *Therefore, the mean shifting is a beneficial normalization for trended series. However, for seasonal series like traffic, only shifting the mean may not help improve the forecasting performance*. As shown in the spectrum of traffic series, the mean shifting only contributes to a very small portion of covariance. The reason behind this is that

Table 2. Comparison between the learned normalization method by SIN and the heuristic method RevIN. The forecasting model is FEDformer and we report the results of typical seasonal and trend datasets – Traffic, Solar, and Exchange. Our SIN model can adaptively select better normalization, and achieves better performance, especially on the seasonal datasets.

		Seasonal								Trend			
		Traffic				Solar				Exchange			
	Horizon	96	192	336	720	96	192	336	720	96	192	336	720
MAE	RevIN	0.355	0.361	0.363	0.378	0.389	0.388	0.448	0.435	**0.275**	0.373	0.488	**0.779**
	SIN	**0.323**	**0.336**	**0.337**	**0.355**	**0.315**	**0.345**	**0.353**	**0.359**	0.292	**0.372**	**0.466**	0.846
MSE	RevIN	0.609	0.637	0.645	0.688	0.345	0.377	0.461	0.449	**0.145**	0.261	0.437	**1.037**
	SIN	**0.493**	**0.509**	**0.522**	**0.554**	**0.218**	**0.261**	**0.281**	**0.291**	0.148	**0.237**	**0.354**	1.053

although the mean shifting is local invariant it is not globally diverse. In Table 2 of the next section, we also show that our SIN method achieves better performance improvement compared to instance normalization, especially on seasonal series.

SIN produces more "normal" distributions. Additionally, we compare the data distribution after being normalized by the mean shifting operation and our SIN. The distributions are plotted in Figure 4. From the figures, we can see that on the trended series (Exchange), the SIN produces a similar normalized distribution as the mean shifting normalization. However, on the seasonal series (Traffic), SIN produces distributions that are closer to the normal distribution, showing the advantage of our adaptive method.

5.3. Main Results

We report the multivariate forecasting results in Table 1. The PEMS datasets have a forecasting horizon of $H \in \{12, 24, 48, 96\}$ while the others have a forecasting horizon of $H \in \{96, 192, 336, 720\}$. As for the input sequence length, we follow the traditional protocol and fix $L = 96$ for all the models. Full results are provided in Appendix F. As shown in the table, we clearly find that our proposed SIN framework can boost these models by a large margin in most cases of the benchmark dataset. The improvement of the SIN method can be attributed to the reasons analyzed in Section 5.2. SIN automatically selects suitable normalization methods according to the characteristics of the datasets and transforms the data to a more normal distribution. On both trended datasets (e.g., Exchange) and seasonal datasets (Traffic, PEMS), SIN improves all the models' forecasting performance in all cases. On Exchange-Rate, SIN improves the performance to around 0.45 MSE for all the models. It is a **72%** improvement for Informer. On Traffic, SIN improves Autoformer from 0.615 to 0.539 (**12.3%**) and improves FEDformer from 0.605 to 0.519 (**14.2%**). On the four PEMS datasets, SIN also shows a huge improvement over the original Autoformer. On PEMS07, a **70%** improvement is shown.

5.4. Comparison to Heuristic Normalization

In Section 5.2, we have analyzed our SIN method with the heuristic means shifting normalization which simply transfers the mean from the look-back window to the feature prediction. In the analysis, we have omitted the scaling operation in the instance normalization as well as the affine module in RevIN (Kim et al., 2021). In this section, we compare our method SIN with RevIN, which differs mainly in the normalization and denormalization operation. We conduct the experiments on three typical datasets, with two consisting of the seasonal series and one consisting of mainly trend series. The forecasting error is presented at Table 2. This table shows the superiority of our method in adaptively handling different types of series. On the seasonal series, our method can achieve consistently better results over all the horizons. On the solar dataset, SIN outperforms RevIN by a large margin, which is an around **35.5%** improvement. The improvement on Solar is also **20%**. While on the trend dataset (Exchange), the SIN achieves similar results as RevIN since the learned transformation is almost the same as the instance normalization. The superior performance on seasonal datasets and the similar performance on the trend datasets is the empirical evidence of our analysis in Section 5.2.

6. Conclusion

This paper rethinks the important role of normalization for long-term time series forecasting. In this paper, we answer the questions of which statistics should be selected and how to perform normalization effectively. We argue about two properties – local invariance and global variability – that the statistics should be extracted in the normalization. Then we propose the SIN method to learn the normalization method and validate its effectiveness on various kinds of datasets with various datasets.

Acknowledgments

This research was supported by National Science and Technology Major Project (2022ZD0114805), NSFC (61773198, 62376118,61921006), Collaborative Innovation Center of Novel Software Technology and Industrialization.

Impact Statement

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

References

Abdi, H. Partial least squares regression and projection on latent structure regression (pls regression). *Wiley interdisciplinary reviews: computational statistics*, 2(1):97–106, 2010.

Agarwal, A., Shah, D., Shen, D., and Song, D. On robustness of principal component regression. In *NeurIPS*, pp. 9889–9900, 2019.

Bai, S., Kolter, J. Z., and Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR*, abs/1803.01271, 2018.

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Box, G. E. P. and Jenkins, G. M. Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 17(2):91–109, 1968.

Chao, W., Ye, H., Zhan, D., Campbell, M. E., and Weinberger, K. Q. Revisiting meta-learning as supervised learning. *CoRR*, abs/2002.00573, 2020.

Chatterjee, S. Matrix estimation by universal singular value thresholding. 2015.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. In Wu, D., Carpuat, M., Carreras, X., and Vecchi, E. M. (eds.), *SSST@EMNLP*, pp. 103–111. Association for Computational Linguistics, 2014.

Cirstea, R.-G., Micu, D.-V., Muresan, G.-M., Guo, C., and Yang, B. Correlated time series forecasting using multi-task deep neural networks. In *ICKM*, pp. 1527–1530, 2018.

Cleveland, R. B., Cleveland, W. S., McRae, J. E., and Terpenning, I. Stl: A seasonal-trend decomposition. *J. Off. Stat*, 6(1):3–73, 1990.

Cui, Y., Zheng, K., Cui, D., Xie, J., Deng, L., Huang, F., and Zhou, X. METRO: A generic graph neural network framework for multivariate time series forecasting. *Proc. VLDB Endow.*, 15(2):224–236, 2021.

Dagum, E. B. and Bianconcini, S. *Seasonal adjustment methods and real time trend-cycle estimation*. Springer, 2016.

Dayal, B. S. and MacGregor, J. F. Improved pls algorithms. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 11(1):73–85, 1997.

Deng, J., Chen, X., Jiang, R., Song, X., and Tsang, I. W. St-norm: Spatial and temporal normalization for multi-variate time series forecasting. In Zhu, F., Ooi, B. C., and Miao, C. (eds.), *KDD*.

Donoho, D., Gavish, M., and Romanov, E. Screenot: Exact mse-optimal singular value thresholding in correlated noise. *The Annals of Statistics*, 51(1):122–148, 2023.

Fan, W., Wang, P., Wang, D., Wang, D., Zhou, Y., and Fu, Y. Dish-ts: a general paradigm for alleviating distribution shift in time series forecasting. In *AAAI*, volume 37, pp. 7522–7529, 2023.

Franceschi, J., Dieuleveut, A., and Jaggi, M. Unsupervised scalable representation learning for multivariate time series. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *NeurIPS*, pp. 4652–4663, 2019.

Han, L., Ye, H.-J., and Zhan, D.-C. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2024.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hyndman, R. J. and Athanasopoulos, G. *Forecasting: principles and practice*. OTexts, 2018.

Kadiyala, A. and Kumar, A. Multivariate time series models for prediction of air quality inside a public transportation bus using available software. *Environmental Progress & Sustainable Energy*, 33(2):337–341, 2014.

Kardakos, E. G., Alexiadis, M. C., Vagropoulos, S. I., Simoglou, C. K., Biskas, P. N., and Bakirtzis, A. G. Application of time series and artificial neural network models in short-term forecasting of pv power generation. In *2013 48th International Universities' Power Engineering Conference (UPEC)*, pp. 1–6. IEEE, 2013.

Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H., and Choo, J. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *ICLR*, 2021.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017. arXiv: 1412.6980.

Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, pp. 95–104, 2018.

Lim, B. and Zohren, S. Time series forecasting with deep learning: A survey. *CoRR*, abs/2004.13408, 2020.

Lindgren, F., Geladi, P., and Wold, S. The kernel algorithm for pls. *Journal of Chemometrics*, 7(1):45–59, 1993.

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. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *NeurIPS*, 2022a.

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 *ICLR*, 2022b.

Liu, Y., Wu, H., Wang, J., and Long, M. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *NeurIPS*, 2022c.

Liu, Z., Cheng, M., Li, Z., Huang, Z., Liu, Q., Xie, Y., and Chen, E. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *NeurIPS*, 2023.

Lu, S., Ye, H., and Zhan, D. Tailoring embedding function to heterogeneous few-shot tasks by global and local feature adaptors. In *AAAI*, pp. 8776–8783, 2021.

Madsen, H. *Time series analysis*. Chapman and Hall/CRC, 2007.

Morid, M. A., Sheng, O. R. L., and Dunbar, J. Time series prediction using deep learning methods in healthcare. *ACM Transactions on Management Information Systems*, 14(1):1–29, 2023.

Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., and Iosifidis, A. Deep adaptive input normalization for time series forecasting. *TNNLS*, 31(9):3760–3765, 2020.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pp. 8024–8035. 2019.

Petropoulos, F., Apiletti, D., Assimakopoulos, V., Babai, M. Z., Barrow, D. K., Taieb, S. B., Bergmeir, C., Bessa, R. J., Bijak, J., Boylan, J. E., et al. Forecasting: theory and practice. *International Journal of Forecasting*, 38(3): 705–871, 2022.

Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. Deep state space models for time series forecasting. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *NeurIPS*, pp. 7796–7805, 2018.

Trygg, J. and Wold, S. Orthogonal projections to latent structures (o-pls). *Journal of Chemometrics: A Journal of the Chemometrics Society*, 16(3):119–128, 2002.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. S. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *NeurIPS*, pp. 101–112, 2021a.

Wu, X., Zhang, D., Guo, C., He, C., Yang, B., and Jensen, C. S. Autocts: Automated correlated time series forecasting. *Proc. VLDB Endow.*, 15(4):971–983, 2021b.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., and Zhang, C. Connecting the dots: Multivariate time series forecasting with graph neural networks. In Gupta, R., Liu, Y., Tang, J., and Prakash, B. A. (eds.), *SIGKDD*, pp. 753–763, 2020.

Ye, H., Zhan, D., Li, N., and Jiang, Y. Learning multiple local metrics: Global consideration helps. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(7):1698–1712, 2020.

Ye, H., Han, L., and Zhan, D. Revisiting unsupervised meta-learning via the characteristics of few-shot tasks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(3):3721–3737, 2023.

Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In Williams, B., Chen, Y., and Neville, J. (eds.), *AAAI*, pp. 11121–11128, 2023.

Zhao, P., Zhang, Y.-J., Zhang, L., and Zhou, Z.-H. Adaptivity and non-stationarity: Problem-dependent dynamic regret for online convex optimization. *Journal of Machine Learning Research*, 25(98):1 – 52, 2024.

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 *AAAI*, pp. 11106–11115, 2021.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, 2022.

Zhou, Z. Learnability with time-sharing computational resource concerns. *CoRR*, abs/2305.02217, 2023.

A. Datasets Description

In Section 5.1, we have described the datasets used in the experiments. We detail the description plus the link to download them here:

1. **ETT (Electricity Transformer Temperature)** (Zhou et al., 2021) [3] comprises two hourly-level datasets (ETTh) and two 15-minute-level datasets (ETTm). Each dataset contains seven oil and load features of electricity transformers from July 2016 to July 2018.

2. **Traffic**[4] describes the road occupancy rates. It contains the hourly data recorded by the sensors of San Francisco freeways from 2015 to 2016.

3. **Electricity**[5] collects the hourly electricity consumption of 321 clients from 2012 to 2014.

4. **Exchange-Rate** (Lai et al., 2018) [6] collects the daily exchange rates of 8 countries from 1990 to 2016.

5. **Weather** includes 21 indicators of weather, such as air temperature, and humidity. Its data is recorded every 10 min for 2020 in Germany.

6. **ILI** [7] describes the ratio of patients seen with influenza-like illness and the number of patients. It includes weekly data from the Centers for Disease Control and Prevention of the United States from 2002 to 2021.

7. **Solar**-Energy (Lai et al., 2018) records the solar power production of 137 PV plants in 2006, which is sampled every 10 minutes.

8. **PEMS** (Liu et al., 2022a) contains public traffic network data in California collected by 5-minute windows.

Other details of these datasets have been concluded in Table 3.

Table 3. Detailed dataset descriptions. *Channels* denotes the number of channels in each dataset. *Dataset Split* denotes the total number of time points in (Train, Validation, Test) split respectively. *Prediction Length* denotes the future time points to be predicted and four prediction settings are included in each dataset. *Frequency* denotes the sampling interval of time points.

Dataset	Channels	Prediction Length	Dataset Size	Frequency	Domain
ETTh1, ETTh2	7	{96, 192, 336, 720}	(8545, 2881, 2881)	Hourly	Electricity
ETTm1, ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	15min	Electricity
Exchange	8	{96, 192, 336, 720}	(5120, 665, 1422)	Daily	Economy
Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	10min	Weather
ECL	321	{96, 192, 336, 720}	(18317, 2633, 5261)	Hourly	Electricity
Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	Hourly	Transportation
Solar-Energy	137	{96, 192, 336, 720}	(36601, 5161, 10417)	10min	Energy
PEMS03	358	{12, 24, 48, 96}	(15617,5135,5135)	5min	Transportation
PEMS04	307	{12, 24, 48, 96}	(10172,3375,281)	5min	Transportation
PEMS07	883	{12, 24, 48, 96}	(16911,5622,468)	5min	Transportation
PEMS08	170	{12, 24, 48, 96}	(10690,3548,265)	5min	Transportation

[3] https://github.com/zhouhaoyi/ETDataset
[4] http://pems.dot.ca.gov
[5] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[6] https://github.com/laiguokun/multivariate-time-series-data
[7] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

B. Singular Value Thresholding

(Chatterjee, 2015; Agarwal et al., 2019; Donoho et al., 2023) provide solid theoretical foundations for SVT's applicability in denoising and handling missing values. However, the SIN framework's core aim diverges fundamentally from the primary objectives of SVT. The normalization method inherent to SIN, especially when applied in the context of PLS-SVD, does not endeavor to eliminate "noise" in the traditional sense. Instead, SIN strategically identifies and utilizes components that most significantly explain the covariance between a look-back window x and future y, leaving the rest parts to the base forecaster. Unlike SVT, the rest parts encapsulate local dynamics vital for long-term forecasting, which may inadvertently be lost in pursuit of denoising covariance matrices. This nuanced approach distinguishes SIN from SVT, where SVT's focus is predominantly on minimizing noise impact.

To empirically substantiate this distinction, we incorporated the ScreeNOT method (Donoho et al., 2023) within the SIN framework. We meticulously calculate the standard deviation of predictions from the base model. The Informer forecasting results with SIN over 5 runs are reported in Table 4. The comparative results, as detailed in the table provided, decisively illustrate that applying ScreeNOT's thresholding criteria adversely affects forecasting accuracy and increases result variability. This outcome underscores the fundamental difference in the objectives between SIN and conventional SVT applications: SIN aims to retain significant covariance components—including those not strictly considered noise—for enhanced forecasting efficacy. We can also conclude that the standard deviation of the performance is influenced by the normalization method. A suitable normalization method will lead to better average performance as well as smaller variance on the performance.

Table 4. Informer forecasting results with SIN and SIN (ScreeNOT). We report the error bar over 5 runs.

	Informer		+SIN		+SIN(ScreeNOT)	
	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	0.83+0.025	1.092+0.042	**0.518+0.012**	**0.546+0.041**	0.993+0.051	1.677+0.146
ETTh2	1.763+0.061	4.632+0.263	**0.514+0.032**	**0.533+0.016**	1.308+0.136	3.328+0.683
ETTm1	0.699+0.031	0.886+0.062	**0.453+0.013**	**0.465+0.038**	0.77+0.3123	1.113+1.489
ETTm2	0.896+0.053	1.658+0.178	**0.433+0.003**	**0.412+0.014**	1.305+0.2187	3.124+1.01
PEMS03	0.29+0.007	0.189+0.008	**0.238+0.008**	**0.135+0.011**	0.351+0.019	0.264+0.022
PEMS04	0.234+0.004	0.124+0.003	**0.224+0.015**	**0.113+0.018**	0.366+0.015	0.272+0.018
PEMS07	0.264+0.003	0.197+0.002	**0.216+0.004**	**0.131+0.006**	0.295+0.011	0.225+0.013
PEMS08	0.333+0.006	0.335+0.008	**0.275+0.011**	**0.213+0.026**	0.423+0.016	0.477+0.018
Electricity	0.414+0.006	0.329+0.006	**0.315+0.009**	**0.217+0.016**	0.412+0.009	0.344+0.018
Exchange	1.007+0.014	1.632+0.042	**0.467+0.002**	**0.448+0.012**	1.561+0.144	3.952+0.669
Solar	0.252+0.009	0.233+0.011	**0.257+0.007**	**0.226+0.002**	0.279+0.011	0.257+0.003
Traffic	0.429+0.009	0.769+0.012	**0.432+0.014**	**0.684+0.034**	0.456+0.008	0.769+0.021
Weather	0.557+0.016	0.634+0.034	**0.328+0.008**	**0.270+0.018**	0.521+0.060	0.558+0.127

C. Selective DCT Normalization

This paper shows that the Discrete Cosine Transformation (DCT) and Mean Shifting (MS) are special cases of our SIN framework (**Discussion**,Section 4.2). We enriched our ablation study to encompass additional comparisons, including mean shifting (simplified RevIN), Discrete Cosine Transformation (DCT), and SIN. The transformation of DCT is also selected by SIN criterion (Equation (8)). Table 5 highlights the comparative analysis:

Our analysis revealed that while mean shifting can mitigate some forecasting challenges, its normalization impact is somewhat constrained and does not consistently outperform more adaptive methods like DCT (SIN) and SIN. Intriguingly, DCT (SIN) often delivered superior performance, suggesting that perfectly regular patterns might offer more robustness against real-world time series noise than learned transformations. For example, in Figure 3, there are some learned transformations that exhibiting patterns with sharpness (LSV3/RSV3 in Figure 3.(a)). Anyway, the results not only validate the flexibility and effectiveness of our SIN framework but also underscore the critical role of choosing the appropriate normalization strategy to enhance forecasting accuracy.

Table 5. Comparative results among Mean Shifting (MS), Discrete Cosine Transformation (DCT), and SIN. While mean shifting can mitigate some forecasting challenges, its normalization impact is somewhat constrained and does not consistently outperform more adaptive methods like DCT (SIN) and SIN. Intriguingly, DCT (SIN) often delivered superior performance, suggesting that perfectly regular patterns might offer more robustness against real-world time series noise than learned transformations.

	Transformer		+MS		+DCT (SIN)		+SIN	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	0.819	1.039	0.551	0.59	**0.518**	**0.537**	0.521	0.551
ETTh2	1.544	3.813	0.509	0.517	**0.463**	**0.463**	0.498	0.507
ETTm1	0.685	0.834	0.49	0.507	**0.445**	0.452	0.449	**0.449**
ETTm2	0.87	1.536	0.417	0.413	**0.388**	**0.359**	0.417	0.402
PEMS03	0.236	0.135	0.257	0.163	**0.222**	**0.117**	0.234	0.126
PEMS04	0.21	0.106	0.232	0.122	**0.209**	**0.099**	0.211	0.104
PEMS07	0.239	0.175	0.238	0.138	**0.202**	**0.103**	0.21	0.13
PEMS08	0.277	0.264	0.276	0.199	**0.246**	**0.163**	0.259	0.205
Electricity	0.366	0.271	0.32	0.209	0.292	0.19	**0.288**	**0.183**
Exchange	0.89	1.374	0.51	0.505	**0.427**	**0.371**	0.464	0.447
Solar	0.249	0.237	0.264	0.234	0.274	0.231	**0.25**	**0.216**
Traffic	0.357	0.656	0.364	0.592	**0.346**	0.574	0.363	**0.561**
Weather	0.577	0.663	0.412	0.383	**0.308**	**0.271**	0.316	0.279

D. Validation on Other Forecasters

We expanded our experimentation to include a variety of models beyond transformers, specifically incorporating DLinear, TCN, and GRU. Below (in next reply) are the refined results showcasing the effectiveness of SIN across these different architectures. The results are shown in Table 6. These results affirm the SIN framework's broad applicability and efficiency,

Table 6. Comparison results of SIN applied to DLinear, TCN and GRU.

	DLinear		+SIN		TCN		+SIN		GRU		+SIN	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	0.454	0.444	**0.426**	**0.415**	0.861	1.161	**0.637**	**0.753**	0.754	0.935	**0.754**	**1.002**
ETTh2	0.464	0.469	**0.406**	**0.366**	1.204	2.359	**0.532**	**0.586**	1.522	3.242	**1.14**	**2.576**
ETTm1	0.379	0.357	0.381	0.358	0.831	1.071	**0.475**	**0.474**	0.696	0.921	**0.686**	**0.85**
ETTm2	0.403	0.356	**0.32**	**0.264**	1.162	2.047	**0.456**	**0.432**	0.92	1.604	**0.912**	**1.56**
PEMS03	0.358	0.264	0.396	0.345	0.366	0.256	**0.362**	**0.25**	0.24	0.141	**0.238**	**0.136**
PEMS04	0.354	0.264	0.417	0.377	0.422	0.324	**0.367**	**0.266**	0.209	0.107	0.216	0.112
PEMS07	0.373	0.312	0.401	0.351	0.341	0.261	**0.321**	**0.225**	0.234	0.171	**0.199**	**0.107**
PEMS08	0.397	0.357	0.408	0.385	0.676	0.912	**0.548**	**0.641**	0.264	0.25	**0.261**	**0.214**
Electricity	0.264	0.167	**0.261**	**0.166**	0.679	0.758	**0.627**	**0.68**	0.386	0.297	**0.365**	**0.266**
Exchange	0.416	0.342	**0.378**	**0.291**	1.507	3.276	**1.1**	**1.992**	1.383	3.422	**0.903**	**1.22**
Solar	0.398	0.327	**0.396**	**0.32**	0.324	0.255	**0.301**	**0.224**	0.281	0.225	**0.271**	**0.215**
Traffic	0.296	0.434	0.302	0.437	0.73	1.325	**0.679**	**1.189**	0.396	0.708	**0.391**	**0.677**
Weather	0.298	0.245	**0.295**	**0.245**	0.46	0.436	**0.424**	**0.366**	0.661	0.867	**0.495**	**0.502**

not only enhancing performance on complex models like transformers but also on simpler architectures such as DLinear, with the exception of PEMS datasets. In scenarios where DLinear underperforms, it suggests a potential over-simplification through normalization, indicating SIN's nuanced impact based on the model's inherent complexity.

E. Ablation on Local Invariance and Global Variability

In this section, we validate the necessity of selecting proper statistics by local invariance and global variability. We provide the results of learning transformation by solely local invariance loss ($L_{\text{loc\_inv}}$) and global variability loss ($L_{\text{glo\_var}}$). For the local invariance loss, there is no closed-form solution. Therefore, we use a training stage to learn the transformation $\boldsymbol{u}$ and $\boldsymbol{v}$. The global variability loss forms the traditional PCA problems for both $\boldsymbol{X}$ and $\boldsymbol{Y}$, which have closed-form solutions. However, it can not be thresholded by a unified singular value. Therefore, we select the top 1 singular vectors. The results are shown in Table 7, which shows that both local invariance and global variability can improve the performance of the base forecaster by normalization. However, we also note that

Table 7. The performance of SIN with only local invariance and global variability loss. Combining both local invariance and global variability loss will lead to better performance than using either.

	Transformer		$L_{\text{loc\_inv}}$		$L_{\text{glo\_var}}$		$L_{\text{loc\_inv}} + L_{\text{glo\_var}}$ (SIN)	
Datasets	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	0.819	1.039	0.618	0.699	0.585	0.622	**0.532**	**0.567**
ETTh2	1.544	3.813	0.650	0.810	0.522	0.560	**0.507**	**0.522**
ETTm1	0.685	0.834	0.508	0.533	0.508	0.532	**0.464**	**0.459**
ETTm2	0.870	1.536	0.661	0.965	0.476	0.520	**0.425**	**0.417**
PEMS03	**0.236**	0.135	**0.234**	**0.127**	0.349	0.260	**0.236**	**0.128**
PEMS04	**0.210**	0.106	0.213	**0.103**	0.337	0.239	0.212	**0.105**
PEMS07	0.239	0.175	0.228	0.147	0.266	0.166	**0.211**	**0.132**
PEMS08	0.277	0.264	0.274	0.221	0.389	0.372	**0.262**	**0.212**
Electricity	0.366	0.271	0.366	0.266	0.390	0.303	**0.288**	**0.184**
Exchange	0.890	1.374	0.588	0.617	0.590	0.774	**0.485**	**0.464**
Solar	**0.249**	0.237	0.265	0.225	0.308	0.244	0.255	**0.219**
Traffic	**0.357**	0.656	0.363	0.621	0.438	0.829	0.368	**0.561**
Weather	0.577	0.663	0.486	0.524	0.475	0.466	**0.316**	**0.280**

F. Full Results of Long-Term Forecasting

In this section, we provide the full results of all the datasets described in Section 5.1. The full results can be found in Table 8.

G. Discussion and Limitations

Although the SIN method can learn the normalization method automatically and is a model-agnostic method that can be integrated with various kinds of forecasting models, it still has the following limitations now. First, the current SIN method involves only the linear projection of the data to ensure interpretability and reduce the risk of overfitting. However, there may be complex statistics that can not be expressed by a simple linear projection. Second, the choice of the look-back window size in SIN is critical, as it influences the features selected for normalization. An inappropriate window size could either miss important trends (if too small) or include irrelevant data noise (if too large). Currently, the process of determining the optimal window size lacks a systematic method and largely relies on heuristic approaches. Third, the computational complexity of SIN, especially in the context of large-scale datasets with numerous channels, is a notable limitation. As the algorithm needs an extra phase that involves the calculation of the singular vectors, it requires significant computational resources, potentially limiting its applicability in resource-constrained scenarios. Last, the way of computing the best statistics may change due to the non-stationarity in time series. Considering both the global and local properties (Ye et al., 2020) of time series may help design instance-specific normalization methods to improve the performance (Lu et al., 2021).

Accordingly, future research could focus on developing non-linear transformations such as kernel-based methods or neural networks, thereby improving the efficacy of SIN. Besides, developing an adaptive method to automatically determine the optimal look-back window size based on the dataset's characteristics would significantly enhance SIN's usability and accuracy. Furthermore, the idea of learning the normalization method could be extended to scenarios with few (Ye et al., 2023) or streaming samples (Zhao et al., 2024). We believe that the learned normalization by SIN presents certain

Table 8. Full results on the datasets with forecasting models selected as Autoformer, FEDformer, Informer and Transformer.

		Autoformer		+SIN		FEDformer		+SIN		Informer		+SIN		Transformer		+SIN	
		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	96	0.455	0.451	**0.430**	**0.419**	0.416	0.375	**0.413**	**0.372**	0.767	0.939	**0.446**	**0.438**	0.763	0.905	**0.467**	**0.462**
	192	0.475	0.495	**0.455**	**0.457**	**0.443**	0.419	**0.443**	**0.417**	0.786	1.009	**0.500**	**0.525**	0.837	1.080	**0.489**	**0.498**
	336	0.494	0.519	**0.474**	**0.489**	**0.463**	0.451	0.465	**0.448**	0.903	1.233	**0.518**	**0.559**	0.867	1.121	**0.530**	**0.568**
	720	**0.490**	**0.478**	0.521	0.547	**0.499**	**0.484**	0.509	0.490	0.864	1.187	**0.608**	**0.698**	0.810	1.049	**0.643**	**0.742**
	Avg	0.479	0.486	**0.470**	**0.478**	**0.455**	**0.432**	0.458	**0.432**	0.830	1.092	**0.518**	**0.555**	0.819	1.039	**0.532**	**0.567**
ETTh2	96	**0.389**	**0.347**	0.403	0.362	**0.383**	**0.344**	0.412	0.357	1.352	2.932	**0.484**	**0.495**	1.188	2.131	**0.439**	**0.418**
	192	**0.433**	**0.422**	0.457	0.454	**0.442**	**0.435**	0.472	0.453	2.104	6.339	**0.517**	**0.570**	1.907	5.729	**0.475**	**0.493**
	336	**0.466**	**0.453**	0.515	0.511	**0.478**	**0.484**	0.527	0.527	1.928	5.265	**0.561**	**0.600**	1.681	4.477	**0.538**	**0.559**
	720	**0.484**	**0.478**	0.567	0.600	**0.479**	**0.468**	0.593	0.639	1.670	3.991	**0.586**	**0.623**	1.400	2.913	**0.575**	**0.619**
	Avg	**0.443**	**0.425**	0.485	0.482	**0.446**	**0.433**	0.501	0.494	1.763	4.632	**0.537**	**0.572**	1.544	3.813	**0.507**	**0.522**
ETTm1	96	0.446	0.424	**0.405**	**0.383**	0.413	0.368	**0.373**	**0.320**	0.558	0.624	**0.408**	**0.389**	0.585	0.629	**0.411**	**0.379**
	192	0.490	0.536	**0.419**	**0.412**	0.436	0.412	**0.394**	**0.366**	0.617	0.724	**0.439**	**0.441**	0.686	0.833	**0.442**	**0.426**
	336	0.551	0.657	**0.431**	**0.440**	0.468	0.467	**0.418**	**0.405**	0.873	1.213	**0.465**	**0.485**	0.744	0.931	**0.468**	**0.457**
	720	0.575	0.734	**0.480**	**0.524**	0.487	0.507	**0.451**	**0.475**	0.749	0.985	**0.499**	**0.551**	0.727	0.942	**0.537**	**0.572**
	Avg	0.516	0.588	**0.434**	**0.440**	0.451	0.438	**0.409**	**0.392**	0.699	0.886	**0.453**	**0.466**	0.685	0.834	**0.464**	**0.459**
ETTm2	96	**0.309**	**0.224**	0.339	0.253	**0.283**	**0.190**	0.326	0.211	0.473	0.393	**0.325**	**0.235**	0.547	0.556	**0.311**	**0.230**
	192	**0.342**	**0.286**	0.398	0.321	**0.325**	**0.260**	0.402	0.316	0.709	0.817	**0.420**	**0.388**	0.783	1.167	**0.396**	**0.380**
	336	**0.374**	**0.340**	0.445	0.388	**0.361**	**0.321**	0.465	0.399	0.928	1.460	**0.481**	**0.495**	0.828	1.193	**0.457**	**0.457**
	720	**0.420**	**0.426**	0.526	0.516	**0.421**	**0.430**	0.555	0.547	1.476	3.961	**0.509**	**0.529**	1.322	3.230	**0.535**	**0.600**
	Avg	**0.361**	**0.319**	0.427	0.370	**0.347**	**0.300**	0.437	0.368	0.896	1.658	**0.434**	**0.412**	0.870	1.536	**0.425**	**0.417**
PEMS03	12	0.364	0.236	**0.237**	**0.112**	**0.245**	0.121	**0.245**	**0.113**	0.300	0.206	**0.205**	**0.095**	0.201	0.103	**0.197**	**0.084**
	24	0.479	0.398	**0.287**	**0.157**	**0.253**	0.129	0.259	**0.125**	0.272	0.168	**0.222**	**0.114**	0.217	0.116	**0.211**	**0.098**
	48	0.715	0.834	**0.373**	**0.265**	**0.302**	**0.178**	0.312	0.188	0.292	0.187	**0.251**	**0.151**	**0.244**	0.141	**0.244**	**0.130**
	96	0.868	1.172	**0.488**	**0.430**	0.398	0.308	**0.357**	**0.253**	0.295	0.194	**0.278**	**0.184**	**0.283**	**0.180**	0.291	0.199
	Avg	0.606	0.660	**0.346**	**0.241**	0.299	0.184	**0.293**	**0.170**	0.290	0.189	**0.239**	**0.136**	0.236	0.135	**0.236**	**0.128**
PEMS04	12	0.348	0.218	**0.273**	**0.150**	**0.217**	**0.100**	0.222	0.103	0.229	0.120	**0.197**	**0.086**	0.195	0.093	**0.185**	**0.080**
	24	0.532	0.468	**0.327**	**0.205**	0.246	0.121	**0.229**	**0.109**	0.239	0.127	**0.204**	**0.092**	0.205	0.102	**0.197**	**0.089**
	48	0.710	0.788	**0.388**	**0.289**	0.261	**0.135**	**0.260**	**0.135**	0.238	0.128	**0.232**	**0.117**	0.221	0.114	**0.212**	**0.102**
	96	1.004	1.505	**0.378**	**0.277**	0.332	0.207	**0.289**	**0.174**	**0.228**	**0.121**	0.266	0.166	0.252	0.148	**0.220**	**0.114**
	Avg	0.649	0.745	**0.341**	**0.230**	0.264	0.141	**0.250**	**0.130**	0.234	0.124	**0.224**	**0.115**	0.218	0.106	**0.203**	**0.105**
PEMS07	12	0.299	0.166	**0.265**	**0.136**	**0.205**	0.102	0.208	**0.097**	0.267	0.197	**0.192**	**0.102**	0.230	0.166	**0.180**	**0.083**
	24	0.467	0.375	**0.301**	**0.172**	**0.221**	0.111	**0.221**	**0.110**	0.241	0.177	**0.207**	**0.127**	0.234	0.167	**0.193**	**0.102**
	48	0.815	1.056	**0.372**	**0.252**	0.263	0.168	**0.257**	**0.145**	0.271	0.203	**0.233**	**0.157**	0.245	0.183	**0.222**	**0.144**
	96	1.121	1.802	**0.472**	**0.470**	0.303	0.196	**0.279**	**0.187**	0.277	0.211	**0.244**	**0.174**	0.249	**0.183**	**0.248**	0.198
	Avg	0.675	0.850	**0.353**	**0.258**	0.248	0.144	**0.241**	**0.135**	0.264	0.197	**0.219**	**0.140**	0.239	0.175	**0.211**	**0.132**
PEMS08	12	0.472	0.413	**0.423**	**0.402**	0.280	**0.158**	**0.269**	0.172	0.293	0.282	**0.236**	**0.134**	0.236	0.210	**0.218**	**0.121**
	24	0.555	0.560	**0.465**	**0.486**	0.316	**0.197**	**0.300**	0.208	0.317	0.310	**0.271**	**0.179**	0.267	0.245	**0.243**	**0.137**
	48	0.730	0.891	**0.426**	**0.387**	0.360	**0.265**	**0.343**	0.268	0.345	0.346	**0.309**	**0.273**	0.297	0.291	**0.271**	**0.217**
	96	1.013	1.602	**0.584**	**0.722**	**0.421**	0.465	0.427	**0.396**	0.375	0.402	**0.343**	**0.345**	**0.309**	**0.308**	0.316	0.373
	Avg	0.692	0.866	**0.475**	**0.499**	0.344	0.271	**0.335**	**0.261**	0.333	0.335	**0.290**	**0.233**	0.277	0.264	**0.262**	**0.212**
Electricity	96	0.309	0.195	**0.287**	**0.174**	0.303	0.188	**0.279**	**0.168**	0.391	0.300	**0.307**	**0.202**	0.352	0.251	**0.270**	**0.164**
	192	0.321	0.207	**0.293**	**0.188**	0.310	0.196	**0.281**	**0.173**	0.416	0.329	**0.310**	**0.206**	0.368	0.268	**0.278**	**0.176**
	336	0.327	0.213	**0.303**	**0.197**	0.329	0.216	**0.299**	**0.188**	0.420	0.330	**0.318**	**0.215**	0.364	0.270	**0.293**	**0.186**
	720	0.350	0.241	**0.338**	**0.233**	0.347	0.237	**0.313**	**0.205**	0.431	0.357	**0.341**	**0.245**	0.380	0.294	**0.311**	**0.209**
	Avg	0.327	0.214	**0.305**	**0.198**	0.322	0.209	**0.293**	**0.183**	0.414	0.329	**0.319**	**0.217**	0.366	0.271	**0.288**	**0.184**
Exchange	96	**0.274**	0.143	0.280	**0.138**	**0.281**	0.151	0.292	**0.148**	0.771	0.905	**0.259**	**0.127**	0.588	0.563	**0.254**	**0.114**
	192	**0.389**	0.288	0.403	**0.287**	0.382	0.276	**0.372**	**0.237**	0.836	1.101	**0.352**	**0.230**	0.818	1.131	**0.404**	**0.284**
	336	0.512	0.472	**0.465**	**0.356**	0.491	0.445	**0.466**	**0.354**	1.013	1.607	**0.499**	**0.491**	0.959	1.442	**0.467**	**0.364**
	720	0.853	1.173	**0.821**	**0.999**	**0.819**	1.133	0.846	**1.053**	1.409	2.916	**0.780**	**0.999**	1.196	2.359	**0.815**	**1.096**
	Avg	0.509	0.522	**0.492**	**0.445**	0.496	0.506	**0.494**	**0.448**	1.007	1.632	**0.472**	**0.462**	0.890	1.374	**0.485**	**0.464**
Solar	96	0.643	0.696	**0.400**	**0.308**	0.399	0.316	**0.315**	**0.218**	0.233	0.217	**0.223**	**0.188**	0.218	**0.193**	**0.212**	0.200
	192	0.562	0.564	**0.389**	**0.304**	0.407	0.327	**0.345**	**0.261**	0.252	**0.223**	**0.251**	0.224	**0.247**	0.263	0.255	**0.220**
	336	0.724	0.854	**0.401**	**0.311**	0.429	0.368	**0.353**	**0.281**	**0.267**	**0.240**	0.269	0.249	0.270	0.255	**0.263**	**0.229**
	720	0.682	0.795	**0.403**	**0.337**	0.469	0.399	**0.359**	**0.291**	0.278	0.252	**0.267**	**0.244**	0.283	0.237	**0.268**	**0.226**
	Avg	0.653	0.728	**0.398**	**0.315**	0.426	0.353	**0.343**	**0.263**	0.257	0.233	**0.252**	**0.226**	0.255	0.237	**0.249**	**0.219**
Traffic	96	0.378	0.610	**0.337**	**0.516**	0.361	0.579	**0.323**	**0.493**	0.451	0.770	**0.428**	**0.697**	0.359	0.635	**0.352**	**0.536**
	192	0.386	0.617	**0.345**	**0.529**	0.372	0.601	**0.336**	**0.509**	0.443	0.779	**0.435**	**0.689**	0.354	0.637	**0.345**	**0.540**
	336	0.371	0.602	**0.351**	**0.539**	0.378	0.613	**0.337**	**0.522**	0.434	0.756	**0.420**	**0.686**	0.365	0.661	**0.359**	**0.560**
	720	0.382	0.632	**0.372**	**0.573**	0.382	0.626	**0.355**	**0.554**	0.450	0.771	**0.431**	**0.739**	0.394	0.689	**0.372**	**0.607**
	Avg	0.379	0.615	**0.351**	**0.539**	0.373	0.605	**0.338**	**0.519**	0.444	0.769	**0.429**	**0.703**	0.368	0.656	**0.357**	**0.561**
Weather	96	0.294	0.217	**0.269**	**0.208**	0.328	0.240	**0.280**	**0.215**	0.435	0.397	**0.262**	**0.185**	0.453	0.448	**0.258**	**0.201**
	192	0.351	0.298	**0.337**	**0.286**	0.509	0.518	**0.314**	**0.264**	0.486	0.486	**0.309**	**0.237**	0.566	0.629	**0.289**	**0.236**
	336	0.382	0.350	**0.357**	**0.324**	0.573	0.649	**0.329**	**0.293**	0.544	0.599	**0.351**	**0.292**	0.593	0.681	**0.330**	**0.297**
	720	**0.418**	**0.415**	0.419	0.424	0.416	0.405	**0.382**	**0.370**	0.764	1.053	**0.403**	**0.372**	0.696	0.896	**0.389**	**0.385**
	Avg	0.361	0.320	**0.345**	**0.311**	0.456	0.453	**0.326**	**0.286**	0.557	0.634	**0.331**	**0.271**	0.577	0.663	**0.316**	**0.280**

meta-knowledge (Chao et al., 2020) of the time series, and should be helpful for fast learning of forecaster under resource constraint Concerns (Zhou, 2023).