PDETime: Rethinking Long-Term Multivariate Time Series Forecasting from the perspective of partial differential equations

Shiyi Qi [1] **Zenglin Xu** [1] **Yiduo Li** [1] **Liangjian Wen** [2] **Qingsong Wen** [3] **Qifan Wang** [4] **Yuan Qi** [5]

Abstract

Recent advancements in deep learning have led to the development of various models for long-term multivariate time-series forecasting (LMTF), many of which have shown promising results. Generally, the focus has been on historical-value-based models, which rely on past observations to predict future series. Notably, a new trend has emerged with time-index-based models, offering a more nuanced understanding of the continuous dynamics underlying time series. Unlike these two types of models that aggregate the information of spatial domains or temporal domains, in this paper, we consider multivariate time series as spatiotemporal data regularly sampled from a continuous dynamical system, which can be represented by partial differential equations (PDEs), with the spatial domain being fixed. Building on this perspective, we present PDETime, a novel LMTF model inspired by the principles of Neural PDE solvers, following the encoding-integration-decoding operations. Our extensive experimentation across seven diverse real-world LMTF datasets reveals that PDETime not only adapts effectively to the intrinsic spatiotemporal nature of the data but also sets new benchmarks, achieving state-of-the-art results.

1. Introduction

Multivariate time series forecasting plays a pivotal role in diverse applications such as weather prediction (Angryk et al., 2020), energy consumption (Demirel et al., 2012),

---
*Equal contribution [1]Harbin Institute of Technology, Shenzhen [2]Huawei Technologies Ltd [3]Squirrel AI [4]Meta AI [5]Fudan University. Correspondence to: Shiyi Qi <21s051040@stu.hit.edu.cn>, Zenglin Xu <zenglin@gmail.com>, Yiduo Li <lisa_liyiduo@163.com>, Liangjian Wen <wenliangjian1@huawei.com>, Qingsong Wen <qingsongedu@gmail.com>, Qifan Wang <wqfcr@fb.com>, Yuan Qi <qiyuan@fudan.edu.cn>.

healthcare (Matsubara et al., 2014), and traffic flow estimation (Li et al., 2017). In the field of multivariate time series forecasting, two typical approaches have emerged: historical-value-based (Zhou et al., 2021; Wu et al., 2021; Zeng et al., 2023; Nie et al., 2023), and time-index-based models (Woo et al., 2023; Naour et al., 2023). Historical-value-based models predict future time steps utilizing historical observations, $\hat{x}_{t+1} = f_\theta(x_t, x_{t-1}, ...)$, while time-index-based models only use the corresponding time-index features, $\hat{x}_{t+1} = f_\theta(t + 1)$. Historical-value-based models have been widely explored due to their simplicity and effectiveness, making them the state-of-the-art models in the field. However, it is important to acknowledge that multivariate time series data are often discretely sampled from continuous dynamical systems. This characteristic poses a challenge for historical-value-based models, as they tend to capture spurious correlations limited to the sampling frequency (Gong et al., 2017; Woo et al., 2023). Moreover, despite their widespread use and effectiveness, the progress in developing historical-value-based models has encountered bottlenecks (Li et al., 2023), making it challenging to further improve their performance. This limitation motivates the exploration of alternative approaches that can overcome the shortcomings of historical-value-based models and potentially deliver more robust forecasting models.

In recent years, deep time-index-based models have gradually been studied (Woo et al., 2023; Naour et al., 2023). These models intrinsically avoid the aforementioned problem of historical-value-based models, by mapping the time-index features to target predictions in continuous space through implicit neural representations (Tancik et al., 2020; Sitzmann et al., 2020) (INRs). However, time-index-based models are only characterized by time-index coordinates, which makes them ineffective in modeling complex time series data and lacking in exploratory capabilities.

In this work, we propose a novel perspective by treating multivariate time series as spatiotemporal data discretely sampled from a continuous dynamical system, described by partial differential equations (PDEs) with fixed spatial domains as described in Eq 3.

From the PDE perspective, we observe that historical-value-based and time-index-based models can be considered as

simplified versions of Eq 4, where either the temporal or spatial information is removed.

By examining Eq 4, we can interpret historical-value-based models as implicit inverse problems. These models infer spatial domain information, such as the position and physical properties of sensors, from historical observations. By utilizing the initial condition and spatial domain information, these models predict future sequences. Mathematically, historical-value-based models can be formulated as:

$$\hat{x}_\tau = x_{\tau_0} + \int_{\tau_0}^{\tau} f_\theta(x) d\tau \tag{1}$$

where x represents the spatial domain information.

Time-index-based models, on the other hand, can be seen as models that consider only the temporal domain. These models solely rely on time-index coordinates for prediction and do not explicitly incorporate spatial information. In this case, the prediction is formulated as:

$$\hat{x}_\tau = \int_{\tau_0}^{\tau} f_\theta(t_\tau) d\tau \tag{2}$$

Motivated by the limitations of existing approaches and inspired by neural PDE solvers, we introduce PDETime, a PDE-based model for LMTF. PDETime considers LMTF as an Initial Value Problem, explicitly incorporating both spatial and temporal domains and requiring the specification of an initial condition.

Specifically, PDETime first predicts the partial derivative term $E_\theta(x_{his}, t_\tau) = \alpha_\tau \approx \frac{\partial u(x, t_\tau)}{\partial t_\tau}$ utilizing an encoder in latent space. Unlike traditional PDE problems, the spatial domain information of LMT is unknown. Therefore, the encoder utilizes historical observations to estimate the spatial domain information. Subsequently, a numerical solver is employed to calculate the integral term $z_\tau = \int_{\tau_0}^{\tau} \alpha_\tau d\tau$. The proposed solver effectively mitigates the error issue and enhances the stability of the prediction results compared to traditional ODE solvers (Chen et al., 2018). Finally, PDE-Time employs a decoder to map the integral term from the latent space to the spatial domains and predict the results as $\hat{x}_\tau = x_{\tau_0} + D_\phi(z_\tau)$. Similar to time-index-based models, PDETime also utilizes meta-optimization to enhance its ability to extrapolate across the forecast horizon. Additionally, PDETime can be simplified into a historical-value-based or time-index-based model by discarding either the time or spatial domains, respectively.

In short, we summarize the key contributions of this work:

- We propose a novel perspective for LMTF by considering it as spatiotemporal data regularly sampled from PDEs along the temporal domains.

- We introduce PDETime, a PDE-based model inspired by neural PDE solvers, which tackles LMTF as an Initial Value Problem of PDEs. PDETime incorporates encoding-integration-decoding operations and leverages meta-optimization to extrapolate future series.

- We extensively evaluate the proposed model on seven real-world benchmarks across multiple domains under the long-term setting. Our empirical studies demonstrate that PDETime consistently achieves state-of-the-art performance.

2. Related Work

Multivariate Time Series Forecasting. With the progressive breakthrough made in deep learning, deep models have been proposed to tackle various time series forecasting applications. Depending on whether temporal or spatial is utilized, these models are classified into historical-value-based (Zhou et al., 2021; 2022; Zeng et al., 2023; Nie et al., 2023; Zhang & Yan, 2023; Liu et al., 2024; 2022b;a; Wu et al., 2023), and time-index-based models (Woo et al., 2023). Historical-value-based models, predicting target time steps utilizing historical observations, have been extensively developed and made significant progress in which a large body of work that tries to apply Transformer to forecast long-term series in recent years (Wen et al., 2023). Early works like Informer (Zhou et al., 2021) and Long-Trans (Li et al., 2019) were focused on designing novel mechanism to reduce the complexity of the original attention mechanism, thus capturing long-term dependency to achieve better performance. Afterwards, efforts were made to extract better temporal features to enhance the performance of the model (Wu et al., 2021; Zhou et al., 2022). Recent work (Zeng et al., 2023) has found that a single linear channel-independent model can outperform complex transformer-based models. Therefore, the very recent channel-independent models like PatchTST (Nie et al., 2023) and DLinear (Zeng et al., 2023) have become state-of-the-art. In contrast, time-index-based models (Woo et al., 2023; Fons et al., 2022; Jiang et al., 2023; Naour et al., 2023) are a kind of coordinated-based models, mapping coordinates to values, which was represented by INRs. These models have received less attention and their performance still lags behind historical-value-based models. PDETime, unlike previous works, considers multivariate time series as spatiotemporal data and approaches the prediction target sequences from the perspective of partial differential equations (PDEs).

Implicit Neural Representations. Implicit Neural Representations are the class of works representing signals as a continuous function parameterized by multi-layer perceptions (MLPs) (Tancik et al., 2020; Sitzmann et al., 2020) (instead of using the traditional discrete representation). These

neural networks have been used to learn differentiable representations of various objects such as images (Henzler et al., 2020), shapes (Liu et al., 2020; 2019), and textures (Oechsle et al., 2019). However, there is limited research on INRs for times series (Fons et al., 2022; Jiang et al., 2023; Woo et al., 2023; Naour et al., 2023; Jeong & Shin, 2022). And previous works mainly focused on time series generation and anomaly detection (Fons et al., 2022; Jeong & Shin, 2022). DeepTime (Woo et al., 2023) is the work designed to learn a set of basis INR functions for forecasting, however, its performance is worse than historical-value-based models. In this work, we use INRs to represent spatial domains and temporal domains.

Neural PDE Solvers. Neural PDE solvers which are used for temporal PDEs, are laying the foundations of what is becoming both a rapidly growing and impactful area of research. These neural PDE solvers fall into two broad categories, *neural operator methods* and *autoregressive methods*. The neural operator methods (Kovachki et al., 2021; Li et al., 2020; Lu et al., 2021) treat the mapping from initial conditions to solutions as time t as an input-output mapping learnable via supervised learning. For a given PDE and given initial conditions u_0, the neural operator $\mathcal{M}$ is trained to satisfy $\mathcal{M}(t, u_0) = u(t)$. However, these methods are not designed to generalize to dynamics for out-of-distribution t. In contrast, the autoregressive methods (Bar-Sinai et al., 2019; Greenfeld et al., 2019; Hsieh et al., 2019; Yin et al., 2022; Brandstetter et al., 2021) solve the PDEs iteratively. The solution of autoregressive methods at time $t + \Delta t$ as $u(t + \Delta) = \mathcal{A}(u(t), \Delta t)$. In this work, We consider multivariate time series as data sampled from a continuous dynamical system according to a regular time discretization, which can be described by partial differential equations. For the given initial condition x_{τ_0}, we use the numerous solvers (Euler solver) to simulate target time step x_τ.

3. Method

3.1. Problem Formulation

In contrast to previous works (Zhou et al., 2021; Woo et al., 2023), we regard multivariate time series as the spatiotemporal data regularly sampled from partial differential equations along the temporal domain, which has the following form:

$$\mathcal{F}(u, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x^1}, ..., \frac{\partial^2 u}{\partial t^2}, \frac{\partial^2 u}{\partial x^2}, ...) = 0,$$
$$u(x, t) : \Omega \times \mathcal{T} \to \mathcal{V}, \qquad (3)$$

subject to initial and boundary conditions. Here u is a spatio-temporal dependent and multi-dimensional continuous vector field, $\Omega \in \mathbb{R}^C$ and $\mathcal{T} \in \mathbb{R}$ denote the spatial and temporal domains, respectively. For multivariate time series data, we regard channel attributes as spatial domains (e.g., the position and physical properties of sensors). The value
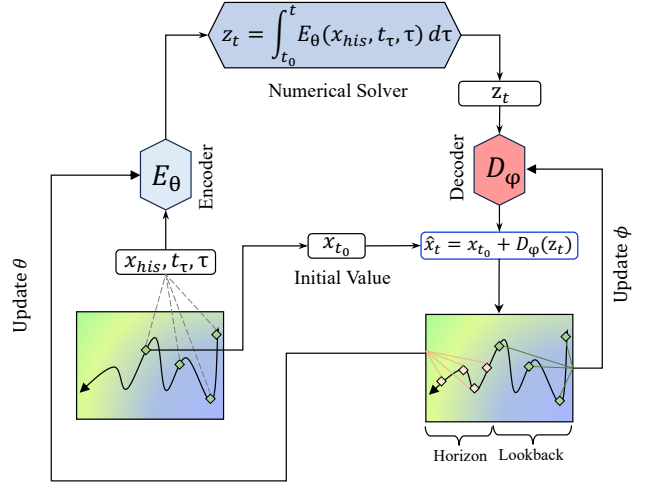

Figure 1. The framework of proposed PDETime which consists of an Encoder E_θ, a Solver, and a Decoder D_ϕ. Given the initial condition x_{τ_0}, PDETime first simulates $\frac{\partial u(x, t_\tau)}{\partial t_\tau}$ at each time step t_τ using the Encoder $E_\theta(x_{his}, t_\tau, \tau)$; then uses the Solver to compute $\int_{\tau_0}^{\tau} \frac{\partial u(x, t_\tau)}{\partial t_\tau} dt_\tau$, which is a numerical solver; finally, the Decoder maps integral term z_τ from latent space to special domains and predict the final results $\hat{x}_\tau = x_{\tau_0} + D_\phi(z_\tau)$.

of the spatial domains, x, is fixed but unknown (e.g., the position of sensors is fixed but unknown), while the value of the temporal domains is known. We treat LMTF as the Initial Value Problems, where $u(x_t, t) \in \mathbb{R}^C$ at any time t is required to infer $u(x_{t'}, t') \in \mathbb{R}^C$.

Consequently, we forecast the target series by the following formula:

$$u(x, t) = u(x, t_0) + \int_{t_0}^{t} \frac{\partial u(x, \tau)}{\partial \tau} d\tau. \qquad (4)$$

DETime initiates its process with a single initial condition, denoted as $u(x, t_0)$, and leverages neural networks to project the system's dynamics forward in time. The procedure unfolds in three distinct steps. Firstly, PDETime generates a latent vector, α_τ of a predefined dimension d, utilizing an encoder function, $E_\theta : \Omega \times \mathcal{T} \to \mathbb{R}^d$ (denoted as the ENC step). Subsequently, it employs an Euler solver, a numerical method, to approximate the integral term, $z_t = \int_{t_0}^{t} \alpha_\tau d\tau$, effectively capturing the system's evolution over time (denoted as the SOL step). In the final step, PDETime translates the latent vectors, z_t, back into the spatial domain using a decoder, $D_\phi : \mathbb{R}^d \to \mathcal{V}$ to reconstruct the spatial function (denoted as the DEC step). This results in the

following model, illustrated in Figure 1,

$$(ENC) \ \alpha_\tau = E_\theta(x_{his}, t_\tau, \tau) \tag{5}$$

$$(SOL) \ z_\tau = \int_{\tau_0}^{\tau} \alpha_\tau d\tau \tag{6}$$

$$(DEC) \ x_\tau = D_\phi(z_\tau). \tag{7}$$

We describe the details of the components in Section 3.2.

3.2. Components of PDETime

3.2.1. ENCODER: $\alpha_\tau = E_\theta(x_{his}, t_\tau, \tau)$

The Encoder computes the latent vector α_τ representing $\frac{\partial u(x, t_\tau)}{\partial t_\tau}$. In contrast to previous works (Chen et al., 2022; Yin et al., 2022) that compute $\frac{\partial u(x, t_\tau)}{\partial t_\tau}$ using auto-differentiation powered by deep learning frameworks, which often suffers from poor stability due to uncontrolled temporal discretizations, we propose directly simulating $\frac{\partial u(x, t_\tau)}{\partial t_\tau}$ by a carefully designed neural networks $E_\theta(x_{his}, t_\tau, \tau)$. The overall architecture of $E_\theta(x_{his}, t_\tau, \tau)$ is shown in Figure 2. As mentioned in Section 3, since the spatial domains of MTS data are fixed and unknown, we can only utilize the historical data $x_{his} \in \mathbb{R}^{H \times C}$ to assist the neural network in fitting the value of spatial domains x. To represent the high frequency of τ, x_{his}, and t_τ, we use concatenated Fourier Features (CFF) (Woo et al., 2023; Tancik et al., 2020) and SIREN (Sitzmann et al., 2020) with k layers.

$$\tau^{(i)} = \text{GELU}(W_\tau^{(i-1)} \tau^{(i-1)} + b_\tau^{(i-1)})$$
$$x_\tau^{(i)} = \sin(W_x^{(i-1)} x_\tau^{(i-1)} + b_x^{(i-1)})$$
$$t_\tau^{(i)} = \sin(W_t^{(i-1)} t_\tau^{(i-1)} + b_t^{(i-1)}), \ i = 1, ..., k \tag{8}$$

where $x_\tau^{(0)} \in \mathbb{R}^{H \times C} = x_{his}$, $t_\tau^{(0)} \in \mathbb{R}^t$ is the temporal feature, and $\tau \in \mathbb{R}$ is the index feature where $\tau_i = \frac{i}{H+L}$ for $i = 0, 1, ..., H + L$, L and H are the look-back and horizon length, respectively. We employ CFF to represent $\tau^{(0)}$, i.e. $\tau^{(0)} = [\sin(2\pi \mathbf{B}_1 \tau), \cos(2\pi \mathbf{B}_1 \tau), ..., \sin(2\pi \mathbf{B}_s \tau), \cos(2\pi \mathbf{B}_s \tau)]$ where $\mathbf{B}_s \in \mathbb{R}^{\frac{d}{2} \times c}$ is sampled from $\mathcal{N}(0, 2^s)$.

After representing $\tau \in \mathbb{R}^d$, $t_\tau \in \mathbb{R}^t$, and $x_\tau \in \mathbb{R}^{d \times C}$ with INRs, the Encoder aggregates x_τ and t_τ using τ, formulated as $f_\theta(\tau, t_\tau, x_\tau)$:

$$\tau = \tau + \text{Attn}(\tau, x_\tau)$$
$$\tau = \tau + \text{Linear}([\tau; t_\tau]) \tag{9}$$

where $[\cdot; \cdot]$ is is the row-wise stacking operation. In this work, we utilize attention mechanism (Vaswani et al., 2017) and linear mapping to aggregate the information of spatial domains and temporal domains, respectively. However, various networks can be designed to aggregate these two types of information.
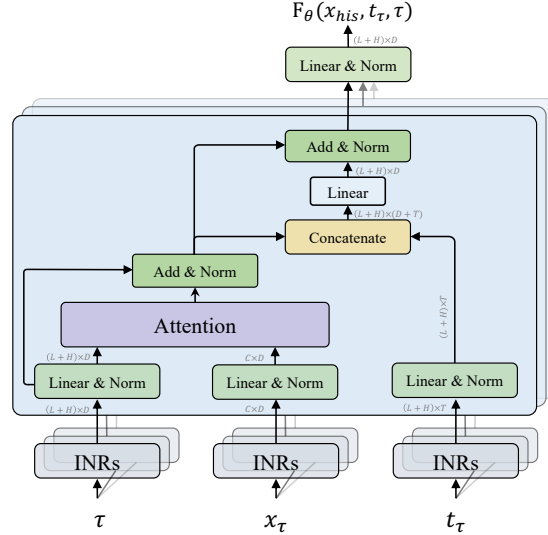

Figure 2. The architecture of the Encoder which is used to simulate $\frac{\partial u(x, t_\tau)}{\partial t_\tau}$. In this work, we use INRs to represent x_{his}, t_τ, and τ, and then use the Attention mechanism and linear mapping to aggregate the information of spatial domains and temporal domains.

Unlike previous works (Chen et al., 2022; Yin et al., 2022), we do not directly make $\alpha_\tau = \frac{\partial u(x, t_\tau)}{\partial t_\tau}$, but rather make $\int_{\tau_0}^{\tau} \alpha_\tau d\tau = \int_{\tau_0}^{\tau} \frac{\partial u(x, t_\tau)}{\partial t_\tau} d\tau$ in latent space, without the need for autoregressive calculation of α_t which can effectively alleviate the error accumulation problem and make the prediction results more stable (Brandstetter et al., 2021).

The pseudocode of the Encoder of PDETime is summarized in Appendix A.3.

3.2.2. SOLVER: $z_t = \int_{\tau_0}^{\tau} \alpha_\tau$

The Solver introduces a numerical solver (Euler Solver) to compute the integral term $z_\tau = \int_{\tau_0}^{\tau} \alpha_\tau d\tau$, which can be approximated as:

$$z_\tau = \int_{\tau_0}^{\tau} \frac{\partial u(x_\tau, t_\tau)}{\partial t_\tau} d\tau \approx \sum_{\tau=\tau_0}^{\tau} \frac{\partial u(x_\tau, t_\tau)}{\partial t_\tau} * \Delta \tau$$

$$\approx \sum_{\tau=\tau_0}^{\tau} E_\theta(x_{his}, t_\tau, \tau) * \Delta \tau \tag{10}$$

where we set $\Delta \tau = 1$ for convenience. However, as shown in Figure 3 (a), directly compute $z_\tau = \sum_{\tau=\tau_0}^{\tau} E_\theta(x_{his}, t_\tau, \tau) * \Delta \tau = \sum_{\tau=\tau_0}^{\tau} \alpha_\tau * \Delta \tau$ through Eq 10 can easily lead to error accumulation and gradient problems (Rubanova et al., 2019; Wu et al., 2022; Brandstetter et al., 2021). To address these issues, we propose a modified approach, as illustrated in Figure 3 (b). We divide the sequence into non-overlapping patches of length S, where $\frac{H+L}{S}$ patches are obtained. For $\tau(H + L) \mod S = 0$, we directly estimate the integral term as $z_\tau = f_\psi(x_{his}, t_\tau, \tau)$
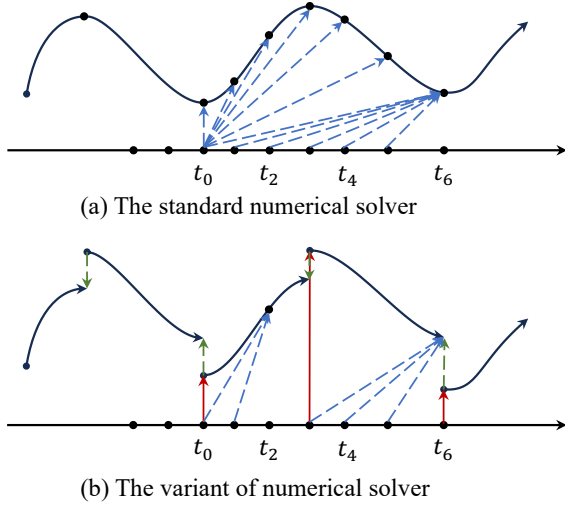
(a) The standard numerical solver

(b) The variant of numerical solver

Figure 3. The framework of standard and variant of numerical solver proposed in our work.

using a neural network f_ψ. Otherwise, we use the numerical solver to estimate the integral term with the integration lower limit $\lfloor \frac{\tau \cdot (H+L)}{S} \rfloor \cdot S$. This modification results in the following formula for the numerical solver:

$$z_t = f_\psi(x_{\tau'}, t_{\tau'}, \tau') + \int_{\tau'}^{\tau} f_\theta(x_\tau, t_\tau, \tau)d\tau,$$

$$\tau' = \lfloor \frac{\tau * (H+L)}{s} \rfloor * S \qquad (11)$$

Furthermore, as shown in Figure 3 (b), Eq 11 breaks the continuity. To address this, we introduce an additional objective function to ensure continuity as much as possible:

$$\mathcal{L}_c = \mathcal{L}(f_\psi(x_\tau, t_\tau, \tau), f_\psi(x_{\tau'}, t_{\tau'}, \tau') + \int_{\tau'}^{\tau} f_\theta(x_\tau, t_\tau, \tau)d\tau)$$

$$\text{s.t. } \tau * (H+L) \mod S = 0, \ \tau' = \frac{\tau * (L+H) - S}{(H+L)}. \qquad (12)$$

The experimental results have shown that incorporating $\mathcal{L}_c$ improves the stability of PDETime.

The pseudocode of the Solver of PDETime is summarized in Appendix A.4.

3.2.3. DECODER: $x_\tau = D_\phi(z_\tau) + x_{\tau_0}$

After estimating the integral term in latent space, we utilize a Decoder D_ϕ to decode z_τ into the spatial function. To tackle the distribution shift problem, we introduce a meta-optimization framework (Woo et al., 2023; Bertinetto et al., 2018) to enhance the extrapolation capability of PDETime. Specifically, we split the time series into look-back window $\mathbf{X}_{t-L:t} = [x_{t-L}, ..., x_t] \in \mathbb{R}^{L \times C}$ and horizon window $\mathbf{Y}_{t+1:t+H} = [x_{t+1}, ..., x_{t+H}] \in \mathbb{R}^{H \times C}$ pairs. We then use

the parameters ϕ and θ to adapt the look-back window and horizon window through a bi-level problem:

$$\theta^* = \arg\min_\theta \sum_{\tau=\tau_{t+1}}^{\tau_{t+H}} \mathcal{L}_p(D_\phi(\int_{\tau_0}^{\tau} E_\theta(x_{his}, t_\tau, \tau)), x_\tau)$$

$$\phi^* = \arg\min_\phi \sum_{\tau=\tau_{t-L}}^{\tau_t} \mathcal{L}_r(D_\phi(\int_{\tau_0}^{\tau} E_\theta(x_{his}, t_\tau, \tau)), x_\tau)$$

$$(13)$$

where $\mathcal{L}_r$ and $\mathcal{L}_p$ denote the reconstruction and prediction loss, respectively (which will be described in detail in Section 3.3). During training, PDETime optimizes both θ and ϕ; while during testing, it only optimizes ϕ to enhance the extrapolation. To ensure speed and efficiency, we employ a single ridge regression for D_ϕ (Bertinetto et al., 2018).

3.3. Optimization

Different from previous works (Zhou et al., 2021; 2022; Wu et al., 2021; Woo et al., 2023) that directly predict the future time steps with the objective $\mathcal{L}_p = \frac{1}{H}\mathcal{L}(x_\tau, \hat{x}_\tau)$. As described as Eq 4, we consider the LMTF as the Initial Value Problem in PDE, and the initial condition x_{τ_0} is known (we use the latest time step in the historical series as the initial condition). Thus the objective of PDETime is to estimate the integral term $\int_{\tau_0}^{\tau} \frac{\partial u(x, t_\tau)}{\partial t_\tau}d\tau$. To achieve this, we define the prediction objective as:

$$\mathcal{L}_p = \frac{1}{H}\mathcal{L}(D_\phi(z_\tau), x_\tau - x_{\tau_0}) \qquad (14)$$

where $z_\tau = \int_{\tau_0}^{\tau} E_\theta(x_{his}, t_\tau, \tau)d\tau$ as described in Section 3.2.1. Additionally, $\mathcal{L}_r$ has the same form $\mathcal{L}_r = \frac{1}{L}(D_\phi(z_\tau), x_\tau - x_{\tau_0})$.

In addition to the prediction objective, we also aim to ensure that the first-order difference of the predicted sequence matches that of the target sequence. Therefore, we introduce the optimization objective:

$$\mathcal{L}_f = \frac{1}{H} \sum_{\tau=\tau_{t+1}}^{\tau_{t+H}} \mathcal{L}(D_\phi(z_\tau) - D_\phi(z_{\tau-1}), x_\tau - x_{\tau-1})$$

$$(15)$$

Combining $\mathcal{L}_p$, $\mathcal{L}_c$, and $\mathcal{L}_f$, the training objective is:

$$\mathcal{L}_p = \mathcal{L}_p + \mathcal{L}_c + \mathcal{L}_f. \qquad (16)$$

4. Experiments

4.1. Experimental Settings

Datasets. We extensively include 7 real-world datasets in our experiments, including four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) (Zhou et al., 2021). Electricity,

Weather and Traffic (Wu et al., 2021), covering energy, transportation and weather domains (See Appendix A.1.1 for more details on the datasets). To ensure a fair evaluation, we follow the standard protocol of dividing each dataset into the training, validation and testing subsets according to the chronological order. The split ratio is 6:2:2 for the ETT dataset and 7:1:2 for the others (Zhou et al., 2021; Wu et al., 2021). We set the length of the lookback series as 512 for PatchTST, 336 for DLinear, and 96 for other historical-value-based models. The experimental settings of DeepTime remain consistent with the original settings (Woo et al., 2023). The prediction length varies in $\{96, 192, 336, 720\}$.

Comparison methods. We carefully choose 10 well-acknowledged models (9 historical-value-based models and 1 time-index-based model) as our benchmarks, including (1) Transformer-based models: Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022), Stationary (Liu et al., 2022b), Crossformer (Zhang & Yan, 2023), PatchTST (Nie et al., 2023), and iTransformer (Liu et al., 2024); (2) Linear-based models: DLinear (Zeng et al., 2023) ; (3) CNN-based models: SCINet (Liu et al., 2022a), TimesNet (Wu et al., 2023); (4) Time-index-based model: DeepTime (Woo et al., 2023). (See Appendix A.1.2 for details of these baselines)

Implementation Details. Our method is trained with the Smooth L1 (Girshick, 2015) loss using the ADAM (Kingma & Ba, 2014) with the initial learning rate selected from $\{10^{-4}, 10^{-3}, 5 \times 10^{-3}\}$. Batch size is set to 32. All experiments are repeated 3 times with fixed seed 2024, implemented in Pytorch (Paszke et al., 2019) and conducted on a single NVIDIA RTX 3090 GPUs. Following Deep-Time (Woo et al., 2023), we set the look-back length as $L = \mu * H$, where μ is a multiplier which decides the length of the look-back windows. We search through the values $\mu = [1, 3, 5, 7, 9]$, and select the best value based on the validation loss. We set layers of INRs $k = 5$ by default, and select the best results from $N = \{1, 2, 3, 5\}$. We summarize the temporal features used in this work in Appendix A.2.

4.2. Main Results and Ablation Study

Comprehensive forecasting results are listed in Table 1 with the best in **Bold** and the second underlined. The lower MSE/MAE indicates the more accurate prediction result. Overall, PDETime achieves the best performance on most settings across seven real-world datasets compared with historical-value-based and time-index-based models. Additionally, experimental results also show that the performance of the proposed PDETime changes quite steadily as the prediction length H increases. For instance, the MSE of PDE-Time increases from 0.330 to 0.365 on the Traffic dataset, while the MSE of PatchTST increases from 0.360 to 0.432, which is the SOTA historical-value-based model. This phenomenon was observed in other datasets and settings as well,

indicating that PDETime retains better long-term robustness, which is meaningful for real-world practical applications.

We perform ablation studies on the Traffic and Weather datasets to validate the effect of **temporal feature** t_τ, **spatial feature** x_{his} and **initial condition** x_{τ_0}. The results are presented in Table 2. **1)** The initial condition x_{t_0} is useful on most settings. As mentioned in Section 3, we treat LMTF as Initial Value Problem, thus the effectiveness of x_{τ_0} validates the correctness of PDETime. **2)** The impact of spatial features x_{his} on PDETime is limited. This may be due to the fact that the true spatial domains x are unknown, and we utilize the historical observations x_{his} to simulate x. However, this approach may not be effective in capturing the true spatial features. **3)** The influence of temporal feature t_τ on PDETime various significantly across different datasets. Experimental results have shown that t_τ is highly beneficial in the Traffic dataset, but its effect on Weather dataset is limited. For example, the period of Traffic dataset may be one day or one week, making it easier for PDETime to learn temporal features. On the other hand, the period of Weather dataset may be one year or longer, but the dataset only contains one year of data. As a result, PDETime cannot capture the complete temporal features in this case.

We conduct an additional ablation study on Traffic to evaluate the ability of different INRs to extract features of x_{his}, t_τ, and τ. In this study, we compared the performance of using the GELU or Tanh activation function instead of sine in SIREN and making $\tau^{(0)} = [\text{GELU}(2\pi\mathbf{B}_1\tau), \text{GELU}(2\pi\mathbf{B}_1\tau), ...]$ or $\tau^{(0)} = [\text{Tanh}(2\pi\mathbf{B}_1\tau), \text{Tanh}(2\pi\mathbf{B}_1\tau), ...]$. Table 4 presents the experimental results, we observe that the sine function (periodic functions) can extract features better than other non-decreasing activation functions. This is because the smooth, non-periodic activation functions fail to accurately model high-frequency information (Sitzmann et al., 2020). Time series data is often periodic, and the periodic nature of the sine function makes it more effective in extracting time series features.

4.3. Effects of Hyper-parameters

We evaluate the effect of four hyper-parameters: look-back window L, number of INRs layers k, number of aggregation layers N, and patch length S on the ETTh1 and ETTh2 datasets. First, we perform a sensitivity on the look-back window $L = \mu * H$, where H is the experimental setting. The results are presented in Table 3. We observe that the test error decreases as μ increases, plateauing and even increasing slightly as μ grows extremely large when the horizon window is small. However, under a large horizon window, the test error increases as μ increases. Next, we evaluate the hyper-parameters N and k on PDETime, as shown in Figure 4 (a) and (b) respectively. We find that the

Table 1. Full results of the long-term forecasting task. We compare extensive competitive models under different prediction lengths following the setting of PatchTST (2023). The input sequence length is set to 336 and 512 for DLinear and PatchTST, and 96 for other historical-value-based baselines. *Avg* means the average results from all four prediction lengths.

Models		PDETime (Ours)		iTransformer (2024)		PatchTST (2023)		Crossformer (2023)		DeepTime (2023)		TimesNet (2023)		DLinear (2023)		SCINet (2022a)		FEDformer (2022)		Stationary (2022b)		Autoformer (2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.292**	**0.335**	0.334	0.368	<u>0.293</u>	0.346	0.404	0.426	0.305	0.347	0.338	0.375	0.299	<u>0.343</u>	0.418	0.438	0.379	0.419	0.386	0.398	0.505	0.475
	192	**0.329**	**0.359**	0.377	0.391	<u>0.333</u>	0.370	0.450	0.451	0.340	0.371	0.374	0.387	0.335	<u>0.365</u>	0.439	0.450	0.426	0.441	0.459	0.444	0.553	0.496
	336	**0.346**	**0.374**	0.426	0.420	0.369	0.392	0.532	0.515	<u>0.362</u>	0.387	0.410	0.411	0.369	<u>0.386</u>	0.490	0.485	0.445	0.459	0.495	0.464	0.621	0.537
	720	**0.395**	**0.404**	0.491	0.459	0.416	0.420	0.666	0.589	<u>0.399</u>	<u>0.414</u>	0.478	0.450	0.425	0.421	0.595	0.550	0.543	0.490	0.585	0.516	0.671	0.561
	Avg	**0.340**	**0.368**	0.407	0.410	0.352	0.382	0.513	0.496	<u>0.351</u>	0.379	0.400	0.406	0.357	<u>0.378</u>	0.485	0.481	0.448	0.452	0.481	0.456	0.588	0.517
ETTm2	96	**0.158**	**0.244**	0.180	0.264	<u>0.166</u>	<u>0.256</u>	0.287	0.366	<u>0.166</u>	0.257	0.187	0.267	0.167	0.260	0.286	0.377	0.203	0.287	0.192	0.274	0.255	0.339
	192	**0.213**	**0.283**	0.250	0.309	<u>0.223</u>	<u>0.296</u>	0.414	0.492	0.225	0.302	0.249	0.309	0.224	0.303	0.399	0.445	0.269	0.328	0.280	0.339	0.281	0.340
	336	**0.262**	**0.318**	0.311	0.348	<u>0.274</u>	<u>0.329</u>	0.597	0.542	0.277	0.336	0.321	0.351	0.281	0.342	0.637	0.591	0.325	0.366	0.334	0.361	0.339	0.372
	720	**0.334**	**0.336**	0.412	0.407	<u>0.362</u>	<u>0.385</u>	1.730	1.042	0.383	0.409	0.408	0.403	0.397	0.421	0.960	0.735	0.421	0.415	0.417	0.413	0.433	0.432
	Avg	**0.241**	**0.295**	0.288	0.332	<u>0.256</u>	<u>0.316</u>	0.757	0.610	0.262	0.326	0.291	0.333	0.267	0.331	0.571	0.537	0.305	0.349	0.306	0.347	0.327	0.371
ETTh1	96	**0.356**	**0.381**	0.386	0.405	0.379	0.401	0.423	0.448	<u>0.371</u>	<u>0.396</u>	0.384	0.402	0.375	0.399	0.654	0.599	0.376	0.419	0.513	0.491	0.449	0.459
	192	**0.397**	**0.406**	0.441	0.436	0.413	0.429	0.471	0.474	<u>0.403</u>	<u>0.420</u>	0.436	0.429	0.405	0.416	0.719	0.631	0.420	0.448	0.534	0.504	0.500	0.482
	336	**0.420**	**0.419**	0.487	0.458	0.435	<u>0.436</u>	0.570	0.546	<u>0.433</u>	<u>0.436</u>	0.491	0.469	0.439	0.443	0.778	0.659	0.459	0.465	0.588	0.535	0.521	0.496
	720	**0.425**	**0.446**	0.503	0.491	<u>0.446</u>	<u>0.464</u>	0.653	0.621	0.474	0.492	0.521	0.500	0.472	0.490	0.836	0.699	0.506	0.507	0.643	0.616	0.514	0.512
	Avg	**0.399**	**0.413**	0.454	0.447	<u>0.418</u>	<u>0.432</u>	0.529	0.522	0.420	0.436	0.458	0.450	0.423	0.437	0.747	0.647	0.440	0.460	0.570	0.537	0.496	0.487
ETTh2	96	**0.268**	**0.330**	0.297	0.349	<u>0.274</u>	<u>0.335</u>	0.745	0.584	0.287	0.352	0.340	0.374	0.289	0.353	0.707	0.621	0.358	0.397	0.476	0.458	0.346	0.388
	192	**0.331**	**0.370**	0.380	0.400	<u>0.342</u>	<u>0.382</u>	0.877	0.656	0.383	0.412	0.402	0.414	0.383	0.418	0.860	0.689	0.429	0.439	0.512	0.493	0.456	0.452
	336	**0.358**	**0.395**	0.428	0.432	<u>0.365</u>	<u>0.404</u>	1.043	0.731	0.523	0.501	0.452	0.452	0.448	0.465	1.000	0.744	0.496	0.487	0.552	0.551	0.482	0.486
	720	**0.380**	**0.421**	0.427	0.445	<u>0.393</u>	<u>0.430</u>	1.104	0.763	0.765	0.624	0.462	0.468	0.605	0.551	1.249	0.838	0.463	0.474	0.562	0.560	0.515	0.511
	Avg	**0.334**	**0.379**	0.383	0.407	<u>0.343</u>	<u>0.387</u>	0.942	0.684	0.489	0.472	0.414	0.427	0.431	0.446	0.954	0.723	0.437	0.449	0.526	0.516	0.450	0.459
ECL	96	**0.129**	**0.222**	0.148	0.240	**0.129**	**0.222**	0.219	0.314	0.137	0.238	0.168	0.272	0.140	0.237	0.247	0.345	0.193	0.308	0.169	0.273	0.201	0.317
	192	**0.143**	**0.235**	0.162	0.253	<u>0.147</u>	<u>0.240</u>	0.231	0.322	0.152	0.252	0.184	0.289	0.153	0.249	0.257	0.355	0.201	0.315	0.182	0.286	0.222	0.334
	336	**0.152**	**0.248**	0.178	0.269	<u>0.163</u>	<u>0.259</u>	0.246	0.337	0.166	0.268	0.198	0.300	0.169	0.267	0.269	0.369	0.214	0.329	0.200	0.304	0.231	0.338
	720	**0.176**	**0.272**	0.225	0.317	<u>0.197</u>	<u>0.290</u>	0.280	0.363	0.201	0.302	0.220	0.320	0.203	0.301	0.299	0.390	0.246	0.355	0.222	0.321	0.254	0.361
	Avg	**0.150**	**0.244**	0.178	0.270	<u>0.159</u>	<u>0.252</u>	0.244	0.334	0.164	0.265	0.192	0.295	0.166	0.263	0.268	0.365	0.214	0.327	0.193	0.296	0.227	0.338
Traffic	96	**0.330**	**0.232**	0.395	0.268	<u>0.360</u>	<u>0.249</u>	0.522	0.290	0.390	0.275	0.593	0.321	0.410	0.282	0.788	0.499	0.587	0.366	0.612	0.338	0.613	0.388
	192	**0.332**	**0.232**	0.417	0.276	<u>0.379</u>	<u>0.256</u>	0.530	0.293	0.402	0.278	0.617	0.336	0.423	0.287	0.789	0.505	0.604	0.373	0.613	0.340	0.616	0.382
	336	**0.342**	**0.236**	0.433	0.283	<u>0.392</u>	<u>0.264</u>	0.558	0.305	0.415	0.288	0.629	0.336	0.436	0.296	0.797	0.508	0.621	0.383	0.618	0.328	0.622	0.337
	720	**0.365**	**0.244**	0.467	0.302	<u>0.432</u>	<u>0.286</u>	0.589	0.328	0.449	0.307	0.640	0.350	0.466	0.315	0.841	0.523	0.626	0.382	0.653	0.355	0.660	0.408
	Avg	**0.342**	**0.236**	0.428	0.282	<u>0.390</u>	<u>0.263</u>	0.550	0.304	0.414	0.287	0.620	0.336	0.433	0.295	0.804	0.509	0.610	0.376	0.624	0.340	0.628	0.379
Weather	96	<u>0.157</u>	<u>0.203</u>	0.174	0.214	**0.149**	**0.198**	0.158	0.230	0.166	0.221	0.172	0.220	0.176	0.237	0.221	0.306	0.217	0.296	0.173	0.223	0.266	0.336
	192	<u>0.200</u>	<u>0.246</u>	0.221	0.254	**0.194**	**0.241**	0.206	0.277	0.207	0.261	0.219	0.261	0.220	0.282	0.261	0.340	0.276	0.336	0.245	0.285	0.307	0.367
	336	**0.241**	**0.281**	0.278	0.296	<u>0.245</u>	<u>0.282</u>	0.272	0.335	0.251	0.298	0.280	0.306	0.265	0.319	0.309	0.378	0.339	0.380	0.321	0.338	0.359	0.395
	720	**0.291**	**0.324**	0.358	0.349	<u>0.314</u>	<u>0.334</u>	0.398	0.418	0.301	0.338	0.365	0.359	0.323	0.362	0.377	0.427	0.403	0.428	0.414	0.410	0.419	0.428
	Avg	**0.222**	**0.263**	0.258	0.279	<u>0.225</u>	<u>0.263</u>	0.259	0.315	0.231	0.286	0.259	0.287	0.246	0.300	0.292	0.363	0.309	0.360	0.288	0.314	0.338	0.382
1st Count		33	33	0	0	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2. Ablation study on variants of PDETime. -Temporal refers that removing the temporal domain feature t_τ; -Spatial refers that removing the historical observations x_{his}; - Initial refers that removing the initial condition x_{τ_0}. The best results are highlighted in **bold**.

| Dataset	Models Metric	PDETime MSE	MAE	-Temporal MSE	MAE	-Spatial MSE	MAE	-Initial MSE	MAE	-Temporal -Spatial MSE	MAE	- All MSE	MAE
Traffic	96	0.330	**0.232**	0.336	0.236	**0.329**	**0.232**	0.334	0.235	0.394	0.268	0.401	0.269
	192	**0.332**	**0.232**	0.368	0.247	0.336	0.234	0.334	**0.232**	0.407	0.269	0.413	0.270
	336	**0.342**	**0.236**	0.378	0.251	0.344	**0.236**	0.343	**0.236**	0.419	0.273	0.426	0.272
	720	**0.365**	**0.244**	0.406	0.265	0.371	0.250	0.368	0.250	0.453	0.291	0.671	0.406
Weather	96	**0.157**	**0.203**	0.158	0.205	0.159	0.205	0.169	0.213	0.159	0.205	0.166	0.212
	192	0.200	0.246	0.206	0.253	**0.198**	**0.243**	0.208	0.248	**0.198**	**0.243**	0.208	0.250
	336	0.241	0.281	**0.240**	0.278	0.246	0.282	0.245	0.287	**0.240**	**0.277**	0.244	0.283
	720	0.291	0.324	0.292	0.323	**0.290**	**0.322**	0.300	0.337	0.294	0.327	0.299	0.337
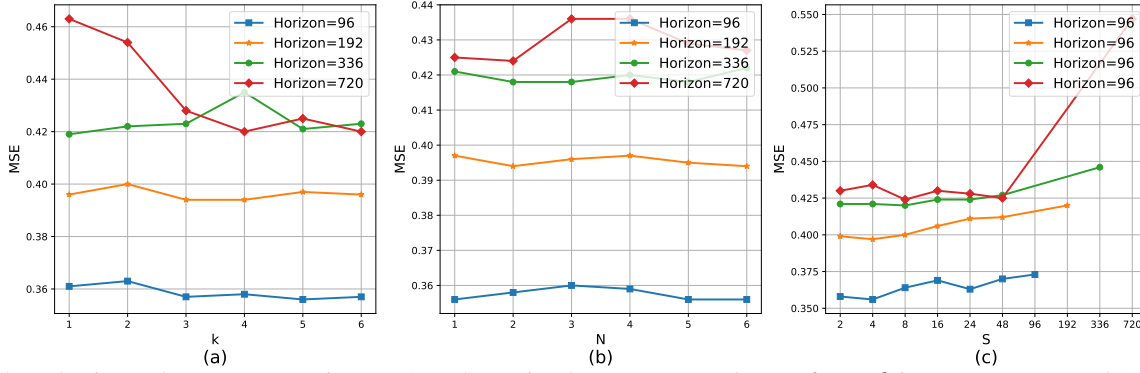
Figure 4. Evaluation on hyper-parameter impact. (a) MSE against hyper-parameter layers of INRs k in Forecaster on ETTh1. (b) MSE against hyper-parameter layers of aggregation module N in Forecaster on ETTh1. (c) MSE against hyper-parameter patch length S in Estimator on ETTh1.

Table 3. Analysis on the look-back window length, based on the multiplier on horizon length, $L = \mu * H$. The best results are highlighted in **bold**.

| Dataset | Horizon | 96 | | 192 | | 336 | | 720 | |
	μ	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	1	0.378	0.386	0.415	0.411	**0.421**	**0.420**	**0.425**	**0.446**
	3	0.359	0.382	**0.394**	**0.404**	0.427	0.421	0.443	0.460
	5	0.360	0.385	0.396	0.405	**0.421**	**0.420**	0.495	0.501
	7	**0.354**	**0.381**	0.398	0.405	0.427	0.429	0.545	0.532
	9	0.356	**0.381**	0.397	0.406	0.446	0.440	1.220	0.882
ETTh2	1	0.288	0.335	0.357	0.381	0.380	0.404	**0.380**	**0.421**
	3	0.276	0.331	0.339	0.374	**0.358**	**0.395**	0.422	0.456
	5	0.275	0.333	**0.331**	**0.370**	0.360	0.408	0.622	0.576
	7	**0.268**	**0.330**	**0.331**	0.378	0.384	0.427	0.624	0.595
	9	0.272	0.331	**0.331**	0.378	0.412	0.451	0.797	0.689

Table 4. Analysis on INRs. PDETime refers to our proposed approach. GELU and Tanh refer to replacing SIREN and CFF with GELU or Tanh activation, respectively. The best results are highlighted in **bold**.

| Dataset | Method | PDETime | | GELU | | Tanh | |
	Metric	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	**0.330**	**0.232**	0.332	0.237	0.338	0.233
	192	**0.332**	**0.232**	0.338	0.241	0.339	0.235
	336	**0.342**	**0.236**	0.348	0.244	0.348	0.238
	720	**0.365**	**0.244**	0.376	0.252	0.366	**0.244**

performance of PDETime remains stable when $k \geq 3$. Additionally, the number of aggregation layers N has a limited impact on PDETime. Furthermore, we investigate the effect of patch length S on PDETime, as illustrated in Figure 4 (c). We varied the patch length from 2 to 48 and evaluate MSE with different horizon windows. As the patch length S increased, the prediction accuracy of PDETime initially improved, reached a peak, and then started to decline. However, the accuracy remains relatively stable throughout. We also extended the patch length to $S = H$. In this case, PDETime performed poorly, indicating that the accumulation of errors has a significant impact on the performance of PDETime. Overall, these analyses provide insights into the effects of different hyper-parameters on the performance of PDETime and can guide the selection of appropriate settings for achieving optimal results.

Due to space limitations, we put extra experiments on robustness and visualization experiments in Appendix A.5 and Appendix A.6, respectively.

5. Conclusion and Future Work

In this paper, we propose a novel LMTS framework PDE-Time, based on neural PDE solvers, which consists of Encoder, Solver, and Decoder. Specifically, the Encoder simulates the partial differentiation of the original function over time at each time step in latent space in parallel. The solver is responsible for computing the integral term with improved stability. Finally, the Decoder maps the integral term from latent space into the spatial function and predicts the target series given the initial condition. Additionally, we incorporate meta-optimization techniques to enhance the ability of PDETime to extrapolate future series. Extensive experimental results show that PDETime achieves state-of-the-art performance across forecasting benchmarks on various real-world datasets. We also perform ablation studies to identify the key components contributing to the success of PDETime.

Future Work Firstly, while our proposed neural PDE solver, PDETime, has shown promising results for long-term multivariate time series forecasting, there are other types of neural PDE solvers that could potentially be applied to this task. Exploring these alternative neural PDE solvers and comparing their performance on LMTF could be an interesting future direction. Furthermore, our ablation studies have revealed that the effectiveness of the historical obser-

vations (spatial domain) on PDETime is limited, and the impact of the temporal domain varies significantly across different datasets. Designing effective network structures that better utilize the spatial and temporal domains could be an important area for future research. Finally, our approach of rethinking long-term multivariate time series forecasting from the perspective of partial differential equations has led to state-of-the-art performance. Exploring other perspectives and frameworks to tackle this task could be a promising direction for future research.

Broader Impacts

Our research is dedicated to innovating time series forecasting techniques to push the boundaries of time series analysis further. While our primary objective is to enhance predictive accuracy and efficiency, we are also mindful of the broader ethical considerations that accompany technological progress in this area. While immediate societal impacts may not be apparent, we acknowledge the importance of ongoing vigilance regarding the ethical use of these advancements. It is essential to continuously assess and address potential implications to ensure responsible development and application in various sectors.

References

Angryk, R. A., Martens, P. C., Aydin, B., Kempton, D., Mahajan, S. S., Basodi, S., Ahmadzadeh, A., Cai, X., Filali Boubrahimi, S., Hamdi, S. M., et al. Multivariate time series dataset for space weather data analytics. *Scientific data*, 7(1):227, 2020.

Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019.

Bertinetto, L., Henriques, J. F., Torr, P. H., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

Brandstetter, J., Worrall, D. E., and Welling, M. Message passing neural pde solvers. In *International Conference on Learning Representations*, 2021.

Chen, P. Y., Xiang, J., Cho, D. H., Chang, Y., Pershing, G., Maia, H. T., Chiaramonte, M., Carlberg, K., and Grinspun, E. Crom: Continuous reduced-order modeling of pdes using implicit neural representations. *arXiv preprint arXiv:2206.02607*, 2022.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Demirel, Ö. F., Zaim, S., Çalişkan, A., and Özuyar, P. Forecasting natural gas consumption in istanbul using neural networks and multivariate time series methods. *Turkish Journal of Electrical Engineering and Computer Sciences*, 20(5):695–711, 2012.

Fons, E., Sztrajman, A., El-Laham, Y., Iosifidis, A., and Vyetrenko, S. Hypertime: Implicit neural representation for time series. *arXiv preprint arXiv:2208.05836*, 2022.

Girshick, R. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

Gong, M., Zhang, K., Schölkopf, B., Glymour, C., and Tao, D. Causal discovery from temporally aggregated time series. In *Uncertainty in artificial intelligence: proceedings of the... conference. Conference on Uncertainty in Artificial Intelligence*, volume 2017. NIH Public Access, 2017.

Greenfeld, D., Galun, M., Basri, R., Yavneh, I., and Kimmel, R. Learning to optimize multigrid pde solvers. In *International Conference on Machine Learning*, pp. 2415–2423. PMLR, 2019.

Henzler, P., Mitra, N. J., and Ritschel, T. Learning a neural 3d texture space from 2d exemplars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8356–8364, 2020.

Hsieh, J.-T., Zhao, S., Eismann, S., Mirabella, L., and Ermon, S. Learning neural pde solvers with convergence guarantees. *arXiv preprint arXiv:1906.01200*, 2019.

Jeong, K.-J. and Shin, Y.-M. Time-series anomaly detection with implicit neural representation. *arXiv preprint arXiv:2201.11950*, 2022.

Jiang, W., Boominathan, V., and Veeraraghavan, A. Nert: Implicit neural representations for general unsupervised turbulence mitigation. *arXiv preprint arXiv:2308.00622*, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*, 2021.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32, 2019.

Li, Y., Yu, R., Shahabi, C., and Liu, Y. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Li, Z., Qi, S., Li, Y., and Xu, Z. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023.

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

Liu, S., Saito, S., Chen, W., and Li, H. Learning to infer implicit surfaces without 3d supervision. *Advances in Neural Information Processing Systems*, 32, 2019.

Liu, S., Zhang, Y., Peng, S., Shi, B., Pollefeys, M., and Cui, Z. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2019–2028, 2020.

Liu, Y., Wu, H., Wang, J., and Long, M. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35:9881–9893, 2022b.

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., and Long, M. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

Matsubara, Y., Sakurai, Y., Van Panhuis, W. G., and Faloutsos, C. Funnel: automatic mining of spatially coevolving epidemics. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 105–114, 2014.

Naour, E. L., Serrano, L., Migus, L., Yin, Y., Agoua, G., Baskiotis, N., Guigue, V., et al. Time series continuous modeling for imputation and forecasting with implicit neural representations. *arXiv preprint arXiv:2306.05880*, 2023.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with

transformers. In *International Conference on Learning Representations*, 2023.

Oechsle, M., Mescheder, L., Niemeyer, M., Strauss, T., and Geiger, A. Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4531–4540, 2019.

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

Rubanova, Y., Chen, R. T., and Duvenaud, D. K. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33: 7537–7547, 2020.

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

Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., and Sun, L. Transformers in time series: A survey. In *International Joint Conference on Artificial Intelligence(IJCAI)*, 2023.

Woo, G., Liu, C., Sahoo, D., Kumar, A., and Hoi, S. Learning deep time-index models for time series forecasting. In *International Conference on Machine Learning*, pp. 37217–37237. PMLR, 2023.

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

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023.

Wu, T., Maruyama, T., and Leskovec, J. Learning to accelerate partial differential equations via latent global evolution. *Advances in Neural Information Processing Systems*, 35:2240–2253, 2022.

Yin, Y., Kirchmeyer, M., Franceschi, J.-Y., Rakotomamonjy, A., and Gallinari, P. Continuous pde dynamics forecasting with implicit neural representations. *arXiv preprint arXiv:2209.14855*, 2022.

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

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

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

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

A. Appendix

In this section, we present the experimental details of PDETime. The organization of this section is as follows:

- Appendix A.1 provides details on the datasets and baselines.

- Appendix A.2 provides details of time features used in this work.

- Appendix A.3 provides pseudocode of the Encoder of PDETime.

- Appendix A.4 provides pseudocode of the Solver of PDETime.

- Appendix A.5 presents the results of the robustness experiments.

- Appendix A.6 visualizes the prediction results of PDETime on the Traffic dataset.

A.1. Experimental Details

A.1.1. DATASETS

we use the most popular multivariate datasets in LMTF, including ETT, Electricity, Traffic and Weather:

- The ETT (Zhou et al., 2021) (Electricity Transformer Temperature) dataset contains two years of data from two separate countries in China with intervals of 1-hour level (ETTh) and 15-minute level (ETTm) collected from electricity transformers. Each time step contains six power load features and oil temperature.

- The Electricity [1] dataset describes 321 clients' hourly electricity consumption from 2012 to 2014.

- The Traffic [2] dataset contains the road occupancy rates from various sensors on San Francisco Bay area freeways, which is provided by California Department of Transportation.

- the Weather [3] dataset contains 21 meteorological indicators collected at around 1,600 landmarks in the United States.

Table 5 presents key characteristics of the seven datasets. The dimensions of each dataset range from 7 to 862, with frequencies ranging from 10 minutes to 7 days. The length of the datasets varies from 966 to 69,680 data points. We split all datasets into training, validation, and test sets in chronological order, using a ratio of 6:2:2 for the ETT dataset and 7:1:2 for the remaining datasets.

Table 5. Statics of Dataset Characteristics.

Datasets	ETTh1	ETTh2	ETTm1	ETTm2	Electricity	Traffic	Weather
Dimension	7	7	7	7	321	862	21
Frequency	1 hour	1 hour	15 min	15 min	1 hour	1 hour	10 min
Length	17420	17420	69680	69680	26304	52696	17544

A.1.2. BASELINES

We choose SOTA and the most representative LMTF models as our baselines, including historical-value-based and time-index-based models, as follows:

- PatchTST (Nie et al., 2023): the current historical-value-based SOTA models. It utilizes channel-independent and patch techniques and achieves the highest performance by utilizing the native Transformer.

- DLinear (Zeng et al., 2023): a highly insightful work that employs simple linear models and trend decomposition techniques, outperforming all Transformer-based models at the time.

---

[1] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014.
[2] http://pems.dot.ca.gov.
[3] https://www.bgc-jena.mpg.de/wetter/.

- Crossformer (Zhang & Yan, 2023): similar to PatchTST, it utilizes the patch technique commonly used in the CV domain. However, unlike PatchTST's independent channel design, it leverages cross-dimension dependency to enhance LMTF performance.

- FEDformer (Zhou et al., 2022): it employs trend decomposition and Fourier transformation techniques to improve the performance of Transformer-based models in LMTF. It was the best-performing Transformer-based model before Dlinear.

- Autoformer (Wu et al., 2021): it combines trend decomposition techniques with an auto-correlation mechanism, inspiring subsequent work such as FEDformer.

- Stationary (Liu et al., 2022b): it proposes a De-stationary Attention to alleviate the over-stationarization problem.

- iTransformer (Liu et al., 2024): it different from pervious works that embed multivariate points of each time step as a (temporal) token, it embeds the whole time series of each variate independently into a (variate) token, which is the extreme case of Patching.

- TimesNet (Wu et al., 2023): it transforms the 1D time series into a set of 2D tensors based on multiple periods and uses a parameter-efficient inception block to analyze time series.

- SCINet (Liu et al., 2022a): it proposes a recursive downsample-convolve-interact architecture to aggregate multiple resolution features with complex temporal dynamics.

- DeepTime (Woo et al., 2023): it is the first time-index-based model in long-term multivariate time-series forecasting.

A.2. Temporal Features

Depending on the sampling frequency, the temporal feature t_τ of each dataset is also different. We will introduce the temporal feature of each data set in detail:

- ETTm and Weather: day-of-year, month-of-year, day-of-week, hour-of-day, minute-of-hour.

- ETTh, Traffic, and Electricity: day-of-year, month-of-year, day-of-week, hour-of-day.

we also normalize these features into [0,1] range.

A.3. Pseudocode

We provide the pseudo-code of Encoder and Solver in Algorithms 1.

Algorithm 1 Pseudocode of the aggregation layer of Encoder

Input: Time-index feature τ, temporal feature t_τ and spatial feature x_{his}.
1: $\tau, t_\tau, x_{his} = \text{Linear}(\tau), \text{Linear}(t_\tau), \text{Linear}(x_{his})$ $\triangleright \tau \in \mathbb{R}^{B \times L \times d}, x_{his} \in \mathbb{R}^{B \times C \times d}, t_\tau \in \mathbb{R}^{B \times L \times t}$
2: $\tau, t_\tau, x_{his} = \text{Norm}(\text{GeLU}(\tau)), \text{Norm}(\sin(t_\tau)), \text{Norm}(\text{GeLU}(x_{his}))$
3: $\tau = \text{Norm}(\tau + \text{Attn}(\tau, x_{his}))$
4: $t_\tau = \text{torch.cat}((\tau,t_\tau),\text{dim=-1})$ $\triangleright t_\tau \in \mathbb{R}^{B \times L \times (d+t)}$
5: $\tau = \text{Norm}(\tau + \text{Linear}(t_\tau))$ $\triangleright \tau \in \mathbb{R}^{B \times L \times d}$
6: **return** τ

A.4. Pseudocode

We provide the pseudo-code of the Solver in Algorithms 2.

Algorithm 2 Pseudocode of the numerical Solver

Input: time-index feature $\tau \in \mathbb{R}^{B \times L \times d}$, batch size B, input length L and dimension d, and patch length S.

1: dudt = MLP (τ) ▷ dudt $\in \mathbb{R}^{B \times L \times d}$

2: u = MLP (τ) ▷ u $\in \mathbb{R}^{B \times L \times d}$

3: u=u.reshape(B, -1, S, d)   ▷ u $\in \mathbb{R}^{B \times \frac{L}{S} \times S \times d}$

4: dudt = -dudt

5: dudt = torch.flip(dudt, [2])

6: dudt = dudt_[:,:,:-1,:]   ▷ dudt $\in \mathbb{R}^{B \times \frac{L}{S} \times (S-1) \times d}$

7: dudt = torch.cat((u[:,:,-1:,:], dudt), dim=-2)   ▷ dudt $\in \mathbb{R}^{B \times \frac{L}{S} \times S \times d}$

8: intdudt = torch.cumsum(dudt, dim=-2)

9: intdudt = torch.flip(dudt, [2])

10: intdudt = intdudt.reshape(B, L , d)   ▷ intdudt $\in \mathbb{R}^{B \times L \times d}$

11: intdudt = MLP(intdudt)   ▷ intdudt $\in \mathbb{R}^{B \times L \times d}$

12: **return** intdudt

A.5. Experimental Results of Robustness

The experimental results of the robustness of our algorithm based on Solver and Initial condition are summarized in Table 6. We also test the effectiveness of continuity loss $\mathcal{L}_c$ in Table 7. The experimental results in Table 6 demonstrate that PDETime can achieve strong performance on the ETT dataset even when using only the Solver or initial value conditions, without explicitly incorporating spatial and temporal information. This finding suggests that PDETime has the potential to perform well even in scenarios with limited data availability. Additionally, we conducted an analysis on the ETTh1 and ETTh2 datasets to investigate the impact of the loss term $\mathcal{L}_c$. Our findings demonstrate that incorporating $\mathcal{L}_c$ into PDETime can enhance its robustness.

Table 6. Analysis on Solver and Initial condition. INRs refers to only using INRs to represent τ; + Initial refers to aggregating initial condition x_{τ_0}; +Solver refers to using numerical solvers to compute integral terms in latent space. The best results are highlighted in **bold**.

Dataset	Method	INRs		INRs+Initial		INRs+Solver	
	Metric	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.371	0.396	0.364	0.384	0.358	0.381
	192	0.403	0.420	0.402	0.409	0.398	0.407
	336	0.433	0.436	0.428	0.420	0.348	0.238
	720	0.474	0.492	0.439	0.452	0.455	0.476
ETTh2	96	0.287	0.352	0.270	0.330	0.285	0.342
	192	0.383	0.412	0.331	0.372	0.345	0.379
	336	0.523	0.501	0.373	0.405	0.357	0.399
	720	0.765	0.624	0.392	0.429	0.412	0.444

Table 7. Analysis on the effectiveness of loss term $\mathcal{L}_c$.

Dataset	Method	PDETime		PDETime-$\mathcal{L}_c$	
	Metric	MSE	MAE	MSE	MAE
ETTh1	96	0.356	0.381	0.357	0.381
	192	0.397	0.406	0.393	0.405
	336	0.420	0.419	0.422	0.420
	720	0.425	0.419	0.446	0.458
ETTh2	96	0.268	0.330	0.271	0.330
	192	0.331	0.370	0.341	0.373
	336	0.358	0.395	0.363	0.397
	720	0.380	0.421	0.396	0.434

A.6. Visualization

We visualize the prediction results of PDETime on the Traffic dataset. As illustrated in Figure 5, for prediction lengths $H = 96, 192, 336, 720$, the prediction curve closely aligns with the ground-truth curves, indicating the outstanding predictive performance of PDETime. Meanwhile, PDETime demonstrates effectiveness in capturing periods of time features.
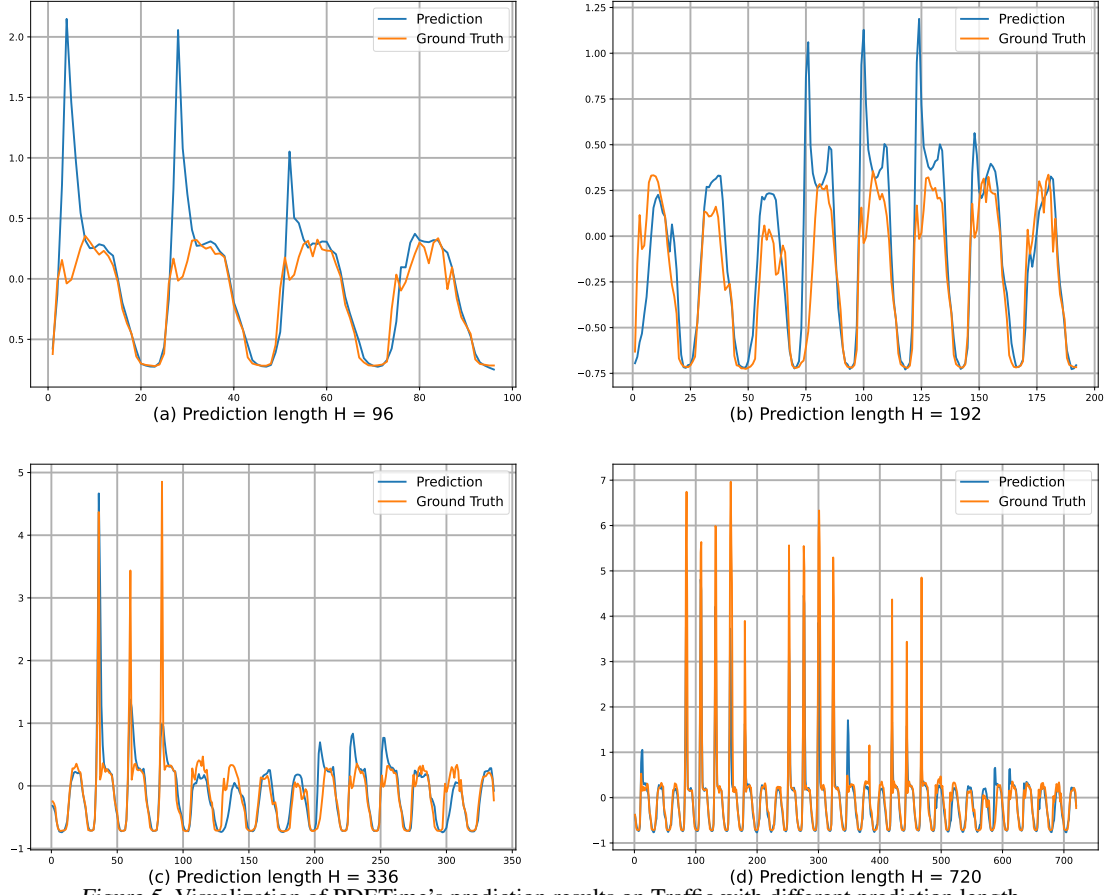

(a) Prediction length H = 96

(b) Prediction length H = 192

(c) Prediction length H = 336

(d) Prediction length H = 720

Figure 5. Visualization of PDETime's prediction results on Traffic with different prediction length.