

# SSD-TS: Exploring the Potential of Linear State Space Models for Diffusion Models in Time Series Imputation

Hongfan Gao  
East China Normal University<sup>1</sup>  
Shanghai, China  
hf.gao@stu.ecnu.edu.cn

Wangmeng Shen  
East China Normal University<sup>1</sup>  
Shanghai, China  
wmshen@stu.ecnu.edu.cn

Xiangfei Qiu  
East China Normal University<sup>1</sup>  
Shanghai, China  
xfqiu@stu.ecnu.edu.cn

Ronghui Xu  
East China Normal University<sup>1</sup>  
Shanghai, China  
rhxu@stu.ecnu.edu.cn

Bin Yang  
East China Normal University<sup>1</sup>  
Shanghai, China  
byang@dase.ecnu.edu.cn

Jilin Hu ✉  
East China Normal University<sup>1,2</sup>  
Shanghai, China  
jlhu@dase.ecnu.edu.cn

## Abstract

Probabilistic time series imputation has been widely applied in real-world scenarios due to its ability for uncertainty estimation and denoising diffusion probabilistic models (DDPMs) have achieved great success in probabilistic time series imputation tasks with its power to model complex distributions. However, current DDPM-based probabilistic time series imputation methodologies are confronted with two types of challenges: 1) *The backbone modules of the denoising parts are not capable of achieving sequence modeling with low time complexity.* 2) *The architecture of denoising modules can not handle the dependencies in the time series data effectively.* To address the first challenge, we explore the potential of state space model, namely Mamba, as the backbone denoising module for DDPMs. To tackle the second challenge, we carefully devise several SSM-based blocks for time series data modeling. Experimental results demonstrate that our approach can achieve state-of-the-art time series imputation results on multiple real-world datasets. Our datasets and code are available at <https://github.com/decisionintelligence/SSD-TS/>

## CCS Concepts

• **Mathematics of computing** → **Time series analysis**; • **Computing methodologies** → *Neural networks*.

## Keywords

Diffusion Models, State Space Models, Time Series Imputation

## ACM Reference Format:

Hongfan Gao, Wangmeng Shen, Xiangfei Qiu, Ronghui Xu, Bin Yang, and Jilin Hu ✉. 2025. SSD-TS: Exploring the Potential of Linear State Space Models for Diffusion Models in Time Series Imputation. In *Proceedings of the*

31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3711896.3737135>

## KDD Availability Link:

The source code of this paper has been made publicly available at <https://doi.org/10.5281/zenodo.15528129>.

## 1 Introduction

The analysis of time series can model the intrinsic patterns within time-series data, thus providing robust support for decision-making in various fields, such as meteorology [29, 44, 46], financial analysis [2, 38, 56, 80], healthcare [48, 55, 59], power systems [74, 79, 90, 92, 93] and traffic [23, 40, 73, 81, 82]. To enhance the reliability of analytical outcomes, it is critical to ensure the integrity of time series. However, due to various reasons such as device failures, human errors, and privacy protection, time series data can easily be incomplete with missing observations at different timestamps.

Time series imputation methods aim to estimate the values of missing points based on the observed points in incomplete time series, thereby restoring the integrity of the time series while preserving its original statistical properties. According to the ability to provide uncertainty of estimations, time series imputation methods can be categorized into the following two perspectives: 1) *Deterministic* [3, 8, 15], and 2) *Probabilistic* [6, 30, 41] imputation methods. *Probabilistic* time series imputation is particularly important in dealing with complex and uncertain data environments, as it provides a quantification of uncertainty for the imputations. The key to probabilistic imputation lies in modeling the posterior distribution. Existing probabilistic time series imputation methods include *Gaussian Process and Variational Autoencoder*-based methods [17], *Normalization Flow*-based methods [64], and *Diffusion*-based methods [72]. Among these, the *Diffusion*-based method has emerged as the optimal choice for probabilistic time series due to their accuracy in posterior modeling and adaptability to different scenarios and various types of time series data.

When selecting a denoising backbone in the diffusion model, the following two key factors need to be considered: 1) **Model compatibility**, and 2) **Time complexity**. Model compatibility involves two key aspects: 1) the backbone of the model should be capable of handling input data effectively. 2) the backbone of the model should align with the model's intended objective (*i.e.*, in

<sup>1</sup>School of Data Science and Engineering.

<sup>2</sup>Engineering Research Center of Blockchain Data Management, Ministry of Education.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

KDD '25, Toronto, ON, Canada

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 979-8-4007-1454-2/2025/08  
<https://doi.org/10.1145/3711896.3737135>

**Table 1: Comparison of our method and existing methods in modeling dependencies and time complexity.**

| Backbone Model            | CNN            | Transformer    | SSM            | SSD-TS (Ours) |
|---------------------------|----------------|----------------|----------------|---------------|
| Global Dependency         | Local          | Global         | Partial        | Global        |
| Time Complexity           | $O(L)$         | $O(L^2)$       | $O(L)$         | $O(L)$        |
| Channel Dependecny        | Independent    | Independent    | Independent    | Dependent     |
| Inter-sequence Dependency | Unidirectional | Unidirectional | Unidirectional | Bidirectional |

diffusion models, the backbone must be capable of modeling noise in the diffusion process). Specifically, the missing observations in time series have correlations with their neighbors on both sides, so it is crucial to design a model by considering information from neighbors of both sides. Moreover, it is also essential to accurately capture the properties of time series, such as global dependencies and channel correlations. Three mainstream denoising backbones are widely used in diffusion models for time series imputation: 1) *Convolutional Neural Networks (CNNs)*-based, 2) *Transformer*-based and 3) *State-Space Model (SSM)*-based backbones. Given a time series with a length of  $L$ , the *CNNs-based backbone* can capture partial information from the neighbors within the receptive fields and has  $O(L)$  time complexity. The *transformer-based backbone* can model temporal dependencies across the entire time series but is with quadratic time complexity  $O(L^2)$ . The *SSM backbone* has a linear time complexity,  $O(L)$ , but it falls short in capturing the information from one side of the neighbor. Moreover, all these backbones fail to capture the channel dependencies in time series. The comparison results of existing backbones and our method in terms of various dependencies and time complexity are presented in Table.1.

Recently, Mamba [12], a linear sequence modeling approach based on state-space models has demonstrated superior performance in various time series tasks, including time series forecasting, imputation and time series foundation models [42, 76]. Compared to Transformer-based models, Mamba-based models have shown stronger sequence modeling capabilities and better performance in time series tasks. However, there has not been research investigating the Mamba model with diffusion models for time series applications. Whether the Mamba model can serve as the backbone of a diffusion model and achieve competitive performance remains an open question.

In this paper, we investigate how to apply Mamba as backbones for time series diffusion models. As shown in Table.1, existing backbones exhibit issues in sequence dependency modeling, time complexity, and inter-channel dependency modeling. To address time complexity, we adopt Mamba of linear complexity, as the foundation for our base module. To tackle intra-sequence and bidirectional dependency modeling, we propose the Bidirectional Attention Mamba (BAM) block, a bidirectional module with temporal attention based on the linear state-space model Mamba, achieving effective and efficient intra-channel dependency modeling. Furthermore, for inter-channel dependencies, we analyze the inductive biases inherent in inter-channel relationships and compare how these biases are captured by the Mamba-based channel modeling module versus CNN- and transformer-based approaches. Our findings indicate that the Mamba-based module provides a more accurate and comprehensive representation, making it a superior choice. Based

on this analysis, we design the Channel Mamba Block (CMB) to effectively model inter-channel dependencies.

Our contributions are summarized as follows:

- (1) We propose SSD-TS, a **State Space Diffusion** model for **Time Series** imputation. It integrates mamba-based blocks as diffusion backbones and equips the model with the capability of probabilistic time series imputation with linear time and space complexity.
- (2) We give a brief analysis of the characteristics and inductive biases of both intra-channel and inter-channel dependencies, thereby empirically elucidating the superiority of employing mamba modules as a backbone for diffusion models. Based on these analyses, we propose bidirectional attention mamba (BAM) block and channel mamba block (CMB) to achieve effective modeling of these dependencies.
- (3) We conduct experiments on multiple real-world datasets for time series imputation task. Our approach achieves state-of-the-art performance across several datasets, different missing scenarios and missing ratios, which demonstrate the effectiveness of our proposed method.

The rest of our paper is organized as follows: In Section 2, we present preliminaries of state space models and diffusion models as well as the problem formulation. In Section 3, we give a brief introduction about the diffusion models and the architecture details about our model. In Section 4, we present and analyze the experiment results. In Section 5, we summarize the related work and in Section 6, we conclude our paper.

## 2 Preliminaries

### 2.1 State Space Models

State Space Models (SSMs) are an emerging approach to model sequential data, which is implemented by finding out state representations to model the relationship between input and output sequences. An SSM receives a one-dimensional sequence  $X \in \mathbb{R}^L$  as the input and outputs a corresponding sequence  $Y \in \mathbb{R}^M$ . Under continuous settings, the SSMs are defined according to Eq.1:

$$\begin{cases} \dot{h}(t) &= \mathbf{A}h(t) + \mathbf{B}x(t) \\ y(t) &= \mathbf{C}h(t) + \mathbf{D}x(t), \end{cases} \quad (1)$$

where  $x(t) \in \mathbb{R}^L$ ,  $y(t) \in \mathbb{R}^M$ ,  $h(t)$ , and  $\dot{h}(t) \in \mathbb{R}^N$  stands for the input, output, hidden state, and derivative of hidden state at timestamp  $t$ , respectively;  $\mathbf{A} \in \mathbb{R}^{N \times N}$ ,  $\mathbf{B} \in \mathbb{R}^{N \times L}$ ,  $\mathbf{C} \in \mathbb{R}^{M \times N}$  and  $\mathbf{D} \in \mathbb{R}^{M \times L}$  are learnable model parameters.

In real-world applications, the input sequences are discrete samplings of continuous sequences. According to [22], under discrete settings, by applying the zero-order hold technique to Eq.1, it can be reformulated as follows.

$$\begin{cases} h_k = \bar{\mathbf{A}}h_{k-1} + \bar{\mathbf{B}}x_k \\ y_k = \mathbf{C}h_k \end{cases}, \quad (2)$$

where  $\bar{\mathbf{A}} = \exp(\Delta\mathbf{A})$ ,  $\bar{\mathbf{B}} = (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A}) - \mathbf{I}) \cdot (\Delta\mathbf{B})$  and  $\Delta$  is the learnable step size in discrete sampling. We can see from Eq.2 that the hidden state is updated according to the input  $x(t)$  and last hidden state  $h(t-1)$  while the output is generated by the hidden

state  $h(t)$  and the input  $x(t)$  and in [21], where it introduces High-order Polynomial Projection Operator (Hippo) to achieve longer sequence modeling.

However, it is worth noticing that  $A, B, C, D$  in Eq.1 and Eq.2 are time-invariant parameters, *i.e.*, they are data-independent parameters and do not change over time. Therefore the model is not capable of assigning different weights at different positions in the input sequence while receiving new inputs. To address this issue, [20] proposed Mamba, in which the parameter matrices  $A, B, C, D$  are input-dependent, thus enhancing the performance of sequence modeling. To tackle the problem of non-parallelization, [20] also introduced selective scan mechanism for effective computing. For further performance and efficiency improvements, [12] point out that SSMs can be categorized as a variant of linear attention model. In this work, we follow the same architecture of parallel Mamba Blocks as [12] and a RMS-norm [88] module is added after the parallel Mamba block. The details of the post-normalization Mamba Block (PNM Block) are illustrated in Fig.3a.

## 2.2 Diffusion Models

Let  $x_t$  be a sequence of variables for  $t = 1, 2, \dots, T$ . The diffusion process consists of two processes: 1) **The forward process** without learnable parameters, which transforms the data distribution into a standard Gaussian distribution by gradually adding noise to the data. 2) **The reverse process** with learnable parameters, which first samples from the standard Gaussian distribution and then progressively denoises the data to approximate the data distribution. The reverse process of diffusion models a parameterized distribution  $p_\theta$  defined with the following Markov chain to approximate the real data distribution:

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t), \quad (3)$$

where  $x_T \sim \mathcal{N}(0, I)$  denotes the latent variable sampled from standard Gaussian distribution and

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_\theta(x_t, t)I), \quad (4)$$

The loss function of DDPM aims at minimizing the difference between the noise  $\epsilon$  in the forward process and the parameterized noise  $\epsilon_\theta$  in the reverse process:

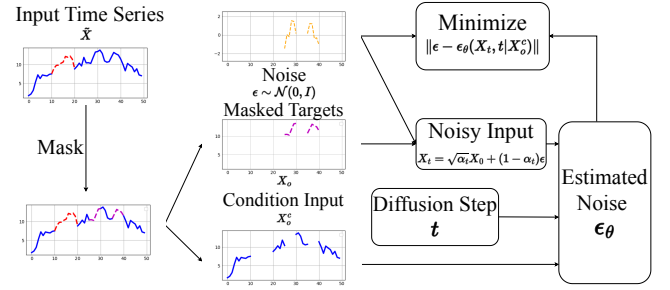
$$\mathcal{L}_d = \mathbb{E}_{x_0, \epsilon} \|\epsilon - \epsilon_\theta(x_t, t)\|, \quad (5)$$

where  $t$  stands for the diffusion time embedding and  $x_t$  is calculated in the forward process.

## 2.3 Problem Formulation

**Definition 2.1 (Time Series).** A time series can be defined as a tuple, denoted as  $\tilde{X} = (X, M, T)$ , where  $X \in \mathbb{R}^{K \times L}$  is the observation matrix with  $K$  observations at a time, which are ordered along  $L$  time intervals chronologically;  $M \in \mathbb{R}^{K \times L}$  is an indicator matrix that indicates whether the observation at  $(i, j)$  in  $X$  is missing or not: if the observation at position  $(i, j)$  is missing, *i.e.*,  $X_{i,j} = \text{NA}$ , then  $M_{i,j} = 1$ , otherwise,  $M_{i,j} = 0$ ;  $T \in \mathbb{R}^L$  is the time stamps of the time series.

**Problem Statement (Probabilistic Time Series Imputation).** Given an incomplete time series  $\tilde{X} = (X, M, T)$ , the problem of



**Figure 1: The self-supervised framework and training process of SSD-TS.** First, some of observed values are masked following the same missing pattern as the missing values (in red) to get masked targets ( $X_0$ , in magenta) and the condition input ( $X_0^c$ , in blue). The noisy input is obtained from  $X_0$  and  $\epsilon$  (in orange) sampled from  $\mathcal{N}(0, I)$ . The objective of the network is to minimize the difference between the parameterized noise  $\epsilon_\theta(X_t, t)$  and  $\epsilon$ . Solid lines in each time series represent observed values, while dashed lines represent missing values.

probabilistic time series imputation aims to approximate the real poster distribution  $p(\tilde{X}|\tilde{X})$ . Deep learning based probabilistic time series imputation methods learn an imputation function  $\mathcal{M}_\theta$ , such that

$$\tilde{X} = \mathcal{M}_\theta(\tilde{X}), \quad (6)$$

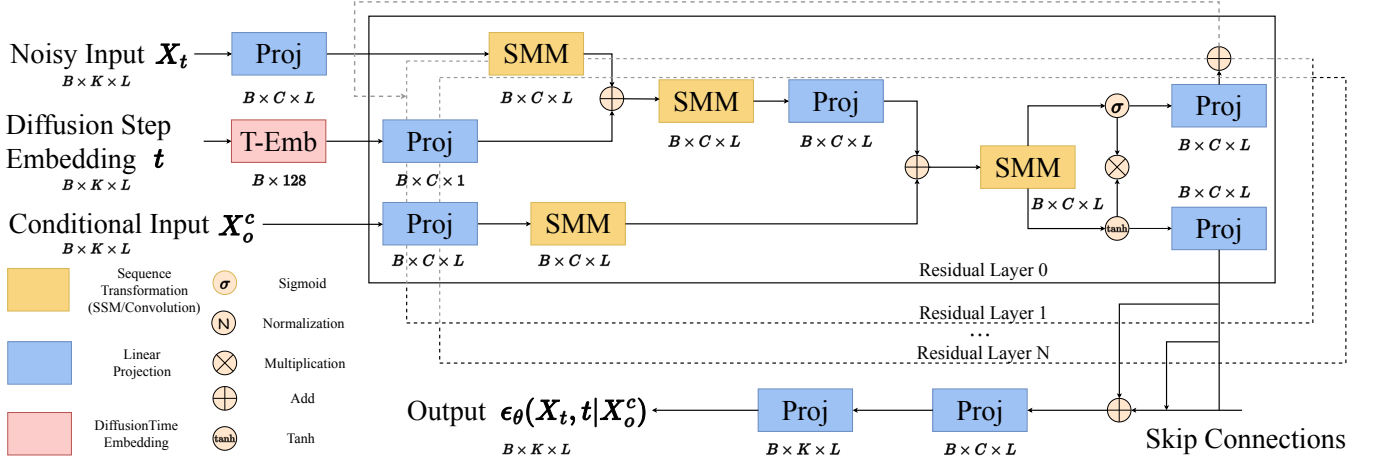
where  $\tilde{X} \in \mathbb{R}^{K \times L}$  is the imputed time series,  $\tilde{X}_{i,j} = \mu_{i,j} \pm \sigma_{i,j}$  denotes the probabilistic output if  $M_{i,j} = 1$ , otherwise  $\tilde{X}_{i,j} = X_{i,j}$ .

## 3 Methodology

### 3.1 Overall Model Architecture

Fig.1 illustrates the overall self-supervised framework and training process of our model. We first mask part of the observed values according to the pattern of missing values, where the masked values serve as the imputation target  $X_0$  during training. The remaining observed values form the conditional input  $X_0^c$  for the noise prediction network  $\epsilon_\theta$ . We then combine  $X_0$  with noise  $\epsilon$  sampled from a standard normal distribution to obtain the noisy input  $X_t$ . Both  $X_0^c$ ,  $X_t$ , and the diffusion step  $t$  are fed into the noise prediction network  $\epsilon_\theta$  to get the parameterized noise.

Fig.2 presents the denoising architecture of our module. The forward process of  $\epsilon_\theta$  are as follows: For each diffusion step, the input consists of the following parts: noisy input  $X_t$ , the condition input  $X_0^c$  and the diffusion step  $t$ . To begin with, the inputs are embedded to the latent diffusion space. The embedding module of noisy inputs and condition inputs share a similar model structure, which consists of a linear projection module followed by an SMM block in Fig.3b. The SMM block is composed of stacks of Bidirectional Attention Mamba (BAM) blocks and Channel Mamba Blocks (CMB), which is introduced in the next part. Due to the relatively limited information from  $t$ , the embedding module of  $t$  only consists of linear projection modules. After the embedding step, the embedded diffusion step is concatenated with the input embeddings. The concatenated embeddings are fed in to a SMM module. Then the output of the SMM module is concatenated with

Figure 2: Architecture of noise estimation module  $\epsilon_\theta$  in SSD-TS.

the condition embeddings. After feeding the final embeddings to another SMM module and final projection module, we can get the noise predictions  $\epsilon_\theta(X_t, t)$ .

### 3.2 Intra-Channel Dependency Modeling

We start by analyzing what the denoising module needs to model within a single channel. Since Triformer [9] first proposes the Patching technique and existing works [54] demonstrate the effectiveness of patching in time series. We do not apply the patching technique to the time series as our approach is non-autoregressive, which means that we generate a complete sequence of size  $L$  for a total of  $T$  steps ( $T < L$ ). The noise described by the denoising module corresponds to the variations between consecutive steps. This necessitates that the denoising module accurately models the internal dependencies within the sequence in order to precisely predict the noise in the intermediate steps.

As shown in Table.1, the main drawback of convolutional neural network (CNN)-based backbones, such as U-Net [65], lies in their inability to capture global information within the sequence. Additionally, stacking convolutional layers increases time complexity. Transformer-based backbones, on the other hand, suffer from a lack of dynamic weight allocation and exhibit quadratic complexity with respect to sequence length  $L$ . Moreover, existing SSM-based models rely on unidirectional intra-channel dependency modeling, limiting their ability to capture global dependencies in time series data.

To address these issues, we design an intra-channel dependency modeling module based on a bidirectional linear state space model. Specifically, given an input representation  $x$ , the BAM module first normalizes the input  $x$  using layer normalization. The normalized features are processed in both forward and backward directions. The forward processing involves a linear convolutional layer followed by a PNM module. The backward sequence shares a similar processing procedure and the input sequence is flipped along the time dimension. After processing in both directions, a temporal attention module is applied to adjust the weights for each time step. Finally, the results from both directions are element-wise summed

after aligning them by position, and a residual connection is added to the original input  $x$ , producing the output of the BAM module. The detailed structure of the BAM module is shown in Fig.3d.

And it is worth noticing the temporal attention module in Fig.3d is not analogous to the attention mechanism used in Transformer. Our temporal attention module takes a sequence as input and applies a convolutional layer followed by a sigmoid activation to produce a set of weights for each position in the sequence. These weights are then multiplied element-wise with the input sequence. The purpose of this module is to adaptively reweight the temporal positions within the sequence, thereby enhancing the modeling of global dependencies and helping the state space model (SSM) better capture long-range temporal relationships.

### 3.3 Inter-Channel Dependency Modeling

Addressing the time series modeling problem, capturing inter-channel dependencies allows for the detection of mutual influences among different channels (variables or features), helping to more accurately reveal the global dynamics and complex interactions of the system, thereby improving predictive performance and generalization ability [62, 78].

For more accurate modeling of the channel dependencies, the key is to identify the inductive biases inherent in the dependencies between channels in time series data. Through an analysis of the characteristics of time series data and previous methods [39, 72], we argue that the modules designed for modeling channel relationships should possess the capability to describe the following inductive biases:

- *Inter-Channel Dependencies*: There are interdependencies among channels and different channels exhibit distinct characteristics.
- *Global Dependencies*: There exist long-term interdependencies and co-evolution among the channels.
- *Local Dependencies*: Certain groups of channels exhibit stronger internal dependencies compared to channels outside of these groups.

- **Dynamic Dependencies:** The dependencies among channels are dynamic. Even for the same time series, the degree of interaction between channels varies on specific downstream tasks.

After identifying the four inductive biases mentioned above, we choose the state space model as the framework for modeling inter-channel dependencies for the following reasons:

- As shown in Eq.1, the state space model updates through the iterative evolution of the hidden state, representing an adaptive updating mechanism. In contrast to the explicit modeling used in the attention structure, it is more accurate when handling complex, long-term inter-channel interactions. Furthermore, while attention relies on the static attention matrix, its ability to describe dynamic dependencies is inferior to that of the state space models.
- Convolution channel modeling suffers from the following limitations: On one hand, the description of local dependencies is influenced by the convolution kernel, which assumes that all dependencies are fixed in space (or time) and cannot capture varying local dependencies. On the other hand, the range of dependencies is constrained by the receptive field, making it incapable of directly and effectively modeling global dependencies.

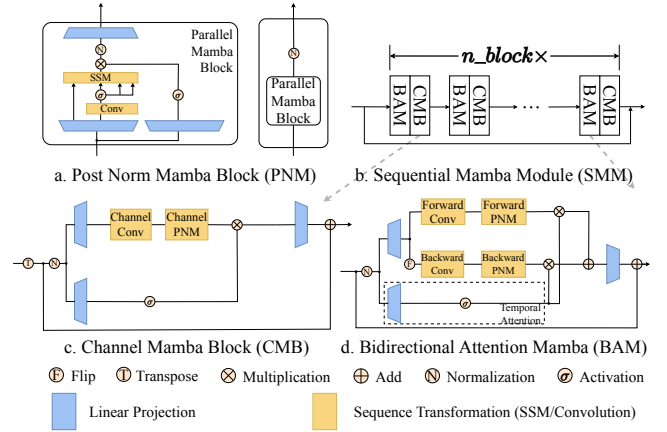
The CMB module takes a latent representation  $x \in \mathbb{R}^{B \times C \times L}$  as input. First, the input representation  $x$  is transposed to  $x^T \in \mathbb{R}^{B \times L \times C}$  for processing along the channel dimension, and the transposed representation is passed through a convolution module and a PNM module. In parallel, a combination of a convolutional layer followed by a sigmoid activation is utilized to dynamically adjust the relative relationships between the channels. After a residual connection with the transposed input  $x^T$ , the CMB module transposes the output back to the original shape of  $x$  to produce the final output. The details of CMB block are shown in Fig.3c.

### 3.4 Diffusion Models for Time Series Imputation

When dealing with time series imputation using diffusion models, consider a time series  $\tilde{X}$ , our goal is to model the posterior  $P(\tilde{X}|X, M, T)$ . To make the modeled posterior more precisely, it is natural to introduce conditions to introduce the diffusion process. Considering the short range and long range inter-dependencies within time series, maximizing the observed values utilized in the diffusion process can effectively improve the performance of the imputation results. On the other hand, due to the fact that all the observed values are utilized as condition inputs in the diffusion process, we do not apply any extra process to the observed values to avoid the error accumulation caused by information propagation, the observed values  $X_0^c$  are condition inputs for the diffusion process. Thus, the reverse process in Eq.3 is modified to a conditional form with time-series inputs:

$$p_\theta(X_{0:T}^m | X_0, X_0^c) = p(X_T^m) \prod_{t=1}^T p_\theta(X_{t-1}^m | X_t^m, X_0^c), \quad (7)$$

where  $X_T^m \sim \mathcal{N}(0, I)$ ,  $X_t^m$  denotes the sequence of latent variables in the diffusion process and  $t \in \{1, 2, \dots, T\}$  is the diffusion time



**Figure 3: Details of PNM, CMB and BAM block in the noise prediction module. (a) PNM: backbone module based on Mamba. (b) SMM: Sequential Mamba module composed of stacks of BAM and CMB. (c) CMB: unidirectional module for inter-channel dependency modeling. (d) BAM: bidirectional module with temporal attention for intra-channel, multi-range dependency modeling.**

steps. Eq.4 is reformulated as:

$$p_\theta(X_{t-1}^m | X_t^m, X_0^c) = \mathcal{N}(X_{t-1}^m; \mu_\theta(X_t^m, t | X_0^c), \sigma_\theta(X_t^m, t | X_0^c)I), \quad (8)$$

Then, the parameterized mean turns to:

$$\mu_\theta(X_t, t) = \frac{1}{\alpha_t} \left( X_t - \frac{\beta_t}{\sqrt{1-\alpha_t}} \epsilon_\theta(X_t, t | X_0^c) \right), \quad (9)$$

where

$$X_t = \sqrt{\alpha_t} X_0 + (1 - \alpha_t) \epsilon, \quad (10)$$

$\{\beta_t \in (0, 1)\}_{t=1}^T$  is a predefined variance scheduler, and  $\alpha_t = \prod_{i=1}^t (1 - \beta_i)$ .

Finally, we get the conditional diffusion loss for the time series imputation task:

$$\mathcal{L} = \mathbb{E}_{X_0, \epsilon} \|\epsilon - \epsilon_\theta(X_t, t | X_0^c)\| = \mathbb{E}_{X_0, \epsilon} \|\epsilon - \epsilon_\theta(\sqrt{\alpha_t} X_0 + (1 - \alpha_t) \epsilon, t | X_0^c)\|, \quad (11)$$

where  $\epsilon \sim \mathcal{N}(0, I)$ .

The network is trained based on Eq.11 to obtain the parameterized noise estimation function  $\epsilon_\theta$ . The missing values are estimated by iterative sampling using  $\epsilon_\theta$ . The training and sampling algorithm is detailed in Alg.1 and Alg.2.

Compared with the attention-based backbones, the advantage of our proposed backbone lies in the following:

- (1) Our proposed is more capable of distinguishing noise and signal. The data is not always made up of signals over the diffusion process. Self-attention mechanism generates attention weights according to input data, i.e.,  $y = \text{softmax} \frac{QK^T}{\sqrt{d}} V$ , which means in the early stage of the diffusion process (data is made up of noise totally) and the intermediate stage of diffusion process (data is made up of noise and signal), the attention weights may be misleading. To avoid this issue, we proposed to use linear state space models (which updates  $h_t = A_t h_{t-1} + B_t x_t$  and outputs  $y_t = C_t^T h_t$  or

**Algorithm 1** Training Procedure of SSD-TS

---

```

1: Input: Observed sequence  $x_0$ , number of iterations  $N$ , variance scheduler  $\beta_t$ 
2: Output: Denoising function  $\epsilon_\theta$ 
3: For  $i = 1$  to  $N$  do:
4:    $t \sim \text{Uniform}(\{1, 2, \dots, T\})$ 
5:    $\epsilon \sim \mathcal{N}(0, I)$ 
6:   Calculate diffusion targets  $x_t$  according to Eq.10
7:   Take gradient step on
        $\nabla_\theta(\|\epsilon - \epsilon_\theta(x_t, t|X_0)\|)$ 
       according to Eq.11
8: End For

```

---

**Algorithm 2** Sampling Procedure of SSD-TS

---

```

1: Input: Trained denoising function  $\epsilon_\theta$ , sampling step  $T$ 
2: Output: Mean prediction  $x_0$ 
3: For  $t = T, T-1, \dots, 1$  do:
4:    $z \sim \mathcal{N}(0, I)$  if  $t > 1$  else  $z = 0$ 
5:    $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( x_t - \frac{1-\alpha_t}{\sqrt{1-\alpha_t}} \right) \epsilon_\theta(x_t, t) + \sigma_t z$ 
6: End For

```

---

$y = \sum_{k=0}^t K_{t-k} x_k$  equivalently) as the backbone diffusion modules, which update parameters independent of content similarity and the gating mechanism of linear SSMs can also help filter noise.

- (2) Our proposed backbone has a controllable frequency response. From a frequency-domain perspective, SSMs offer controllable frequency responses  $\hat{K}(\omega) = C(i\omega I - A)^{-1}B$ , allowing them to suppress broadband noise while preserving signal components concentrated in specific frequency bands. This frequency selectivity enables more robust and adaptive representation learning, whereas attention mechanisms, lacking frequency bias, may fail under similar conditions.

### 3.5 Complexity Analysis

In this part, we will give a brief analysis about the time and space complexity in the SSM module and self-attention module<sup>1</sup>. While dealing with the input sequences, the core component of our module is the PNM module in Fig.3 and the self-attention module in the Transformer architecture, respectively. The time complexity of self-attention module is  $O(CL^2)$  and the space complexity is  $O(L^2 + CL)$ , where  $L$  is the length of the input sequence and  $C$  is the channel of the input sequence.

In our method, the forward process described in Eq.2 is implemented by converting the process to multiplications of structured matrices, which is of time complexity  $O(NCL)$  and of space complexity  $O(CL + N(C + L))$  ( $N$  is a constant number and set as 16 by default). This indicates that our model is of linear time and space complexity with respect to the sequence length  $L$ , which ensures scalability and reduces memory cost for longer sequences.

<sup>1</sup>We do not take the time and space complexity of MLPs before the self-attention module or SSM module into consideration.

## 4 Experiments

### 4.1 Experimental Settings and Evaluation Metrics

All experiments are conducted using PyTorch [58] in Python 3.9 and executed on an NVIDIA RTX3090 GPU. The training process is guided by Eq.11, employing the ADAM optimizer [31] with a learning rate of  $2 \times 10^{-4}$ . The hyperparameters in our experiments can be found in Appendix.7.1.2.

For time series modeling tasks, we explicitly categorize the problem into two paradigms based on the model's capacity to quantify uncertainty, *i.e.*, probabilistic time series modeling and deterministic time series modeling. To evaluate the performance of deterministic modeling methods, we report the Mean Absolute Error (MAE), the Mean Squared Error (MSE), Mean Relative Error (MRE) and Root Mean Squared Error (RMSE) between outputs of the model and ground-truth time series data. As defined in Definition.2.1, the original time series is denoted as  $y \in \mathbb{R}^{K \times L}$ , the output time series is denoted as  $\hat{y} \in \mathbb{R}^{K \times L}$ ,  $M$  is the indicator matrix, the metrics are formulated as follows:

**Mean Absolute Error (MAE):** MAE calculates the average  $L_1$  distance between ground truth and the output values alongside the channel dimension, which is formulated as:

$$\text{MAE}(y, \hat{y}) = \frac{1}{k} \sum_{i=1}^K \sum_{j=1}^L |(y - \hat{y}) \odot (1 - M)|_{i,j} \quad (12)$$

**Mean Square Error (MSE):** MSE calculates the average  $L_2$  between ground truth and the output values alongside the channel dimension, which is formulated as:

$$\text{MSE}(y, \hat{y}) = \frac{1}{k} \sum_{i=1}^K \sum_{j=1}^L ((y - \hat{y}) \odot (1 - M))_{i,j}^2 \quad (13)$$

**Root Mean Square Error (RMSE):** RMSE is the square root of the average  $L_2$  between ground truth and the output values alongside the channel dimension:

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{k} \sum_{i=1}^K \sum_{j=1}^L ((y - \hat{y}) \odot (1 - M))_{i,j}^2} \quad (14)$$

**Mean Relative Error (MRE):** MRE estimates the relative difference between  $y$  and  $\hat{y}$ :

$$\text{MRE}(y, \hat{y}) = \frac{1}{k} \sum_{i=1}^K \sum_{j=1}^L (1 - M)_{i,j} \odot \frac{|(y - \hat{y})|_{i,j}}{y_{i,j}} \quad (15)$$

For probabilistic time series modeling evaluation, we additionally evaluate the Continuous Ranked Probability Score (CRPS). CRPS measures the integral squared difference between the predicted and empirical cumulative distribution functions (CDFs), providing a pointwise evaluation of probabilistic calibration. By applying CRPS, we can effectively evaluate the consistency between the estimated distribution and the real distribution. The CRPS metric is defined as follows:

**Continuous Ranked Probabilistic Score (CRPS):** Given an estimated probability distribution function  $F$  modeled with an observation  $x$ , CRPS evaluates the compatibility and is defined as the

integral of the quantile loss for all quantile levels:

$$\text{CRPS}(F^{-1}, x) = \int_0^1 \Lambda_\alpha(F^{-1}(\alpha, x)) d\alpha, \quad (16)$$

where  $\Lambda_\alpha(q, y) = (\alpha - \mathbf{1}_{y < q})(y - q)$ ,  $\alpha \in [0, 1]$  and  $\mathbf{1}_{y < q}$  the indicator function, i.e., if  $y < q$ , the value of the indicator function is 1, else 0.

Following [72, 83], we separate the interval  $[0, 1]$  to 20 quantile levels with a stepsize of  $s = 0.05$ , and the estimated value of CRPS is:

$$\text{CRPS}(F^{-1}, x) \approx \sum_{i=1}^{19} \frac{2\Lambda_{i \cdot s}(F^{-1}(i \cdot s, x))}{19} \quad (17)$$

For the whole time series  $X \in \mathbb{R}^{K \times L}$ , the CRPS value is normalized for all time steps and channels:

$$\text{CRPS}(F^{-1}, X) = \frac{\sum_{i=1}^K \sum_{j=1}^L \text{CRPS}(F_{i,j}^{-1}, X_{i,j})}{\sum_{i=1}^K \sum_{j=1}^L |X_{i,j}|} \quad (18)$$

## 4.2 Time Series Imputation

**4.2.1 Datasets.** To evaluate the performance of our model on time series imputation tasks, we employ three benchmark datasets, including the MuJoCo dataset [66], the Air Quality dataset (AQI) [84] and Physionet Challenge 2012 dataset [70]. Please refer to Appendix 7.1.1 for more details about the datasets.

It is worth noting that we select these three datasets to include as many potential missing data scenarios as possible. For the Physionet dataset, consistent with previous studies [3, 60, 61, 71], we process the dataset into hourly time series, each containing 48 time steps, with approximately 80% of the data missing. Since the dataset does not include ground truth, we randomly select 10%, 50%, and 90% of the data as ground truth in the test set. For the Air Quality dataset, around 13% of the data is missing in a non-random manner. As this dataset contains structured artificial ground truth, we do not test different missing ratios on the AQI dataset. The MuJoCo dataset, on the other hand, is a complete dataset and we apply hand-crafted masks to simulate masking patterns that are not present in the Physionet and Air Quality datasets.

**4.2.2 Baselines.** Correspondingly, in our experiments, the baselines include both deterministic and probabilistic imputation models. The deterministic models involve: BRITS [3], RDIS [7], SSGAN [47], TIDER [39], and SAITS [15]. The probabilistic imputation models encompass various distribution estimation methods, such as: methods based on variational autoencoders (V-RIN [50]), methods based on diffusion models (CSDI [72], SSSD [1], D<sup>3</sup>M[83], TS-Diff [39]), methods based on Schrödinger Bridge (CSBI [6]), and methods based on Gaussian process (GP-VAE [17]). And on the MuJoCo dataset, we compare our models against RNN GRU-D [4], ODE-RNN [66], NeuralCDE [49], Latent-ODE [66], NAOMI [43], CSDI [72] and SSSD [1] to be consistent with SSSD [1].

**4.2.3 Experimental Results and Analysis.** Table 2 presents the experimental results on the MuJoCo dataset for the random missing (RM) case, with different missing rate scenarios. Under the two higher missing rates (80% and 90%), our method achieves the best performance, with improvements of 48.2% (at 80% missing rate) and 65.8% (at 90% missing rate) compared to the second-best results.

**Table 2: MSE Results on MuJoCo Dataset with missing ratio 70%, 80% and 90% for the missing scenario RM.**

| Model        | 70% RM             | 80% RM                 | 90% RM              |
|--------------|--------------------|------------------------|---------------------|
| RNN GRU-D    | 1.134e-2           | 1.421e-2               | 1.968e-2            |
| ODE-RNN      | 9.86e-3            | 1.209e-2               | 1.647e-2            |
| NeuralCDE    | 8.35e-3            | 1.071e-2               | 1.352e-2            |
| Latent-ODE   | 3.00e-3            | 2.95e-3                | 3.60e-3             |
| NAOMI        | 1.46e-3            | 2.32e-3                | 4.42e-3             |
| NRTSI        | 6.3e-4             | 1.22e-3                | 4.06e-3             |
| CSDI         | <b>2.4e-4±3e-5</b> | <u>6.1e-4±1.0e-4</u>   | 4.84e-3±2e-5        |
| SSSD         | 5.9e-4±8e-5        | 1e-3±5e-5              | <u>1.90e-3±3e-5</u> |
| SSD-TS(Ours) | <u>2.7e-4±1e-5</u> | <b>3.16e-4±9.77e-6</b> | <b>6.5e-4±1e-4</b>  |

**Table 3: RMSE results on Physionet and Air quality dataset. The mean and standard error are obtained by 3 runs and the best result are in bold while the second best results are underlined.**

|                                 | Physionet           |                    |                     | AQI               |
|---------------------------------|---------------------|--------------------|---------------------|-------------------|
|                                 | 10% missing         | 50% missing        | 90% missing         |                   |
| V-RIN                           | 0.628±0.025         | 0.693±0.022        | 0.928±0.013         | 40.11±1.14        |
| BRITS                           | 0.619±0.018         | 0.701±0.021        | 0.847±0.021         | 24.28±0.65        |
| SSGAN                           | 0.607±0.034         | 0.758 ±0.025       | 0.830±0.009         | -                 |
| RDIS                            | 0.635±0.018         | 0.747 ±0.013       | 0.922±0.018         | 37.25±0.31        |
| CSDI                            | 0.531±0.009         | 0.668±0.007        | 0.834±0.006         | 19.21±0.13        |
| CSBI                            | 0.547±0.019         | 0.649 ±0.009       | 0.837±0.012         | 19.07±0.18        |
| SSSD                            | 0.459±0.001         | 0.632±0.004        | 0.824±0.003         | 18.77±0.08        |
| TS-Diff                         | 0.523±0.015         | 0.679±0.009        | 0.845±0.007         | 19.06±0.14        |
| SAITS                           | 0.461±0.009         | 0.636±0.005        | 0.819±0.002         | 18.68±0.13        |
| D <sup>3</sup> M(Constant-Sqrt) | <u>0.438±0.003</u>  | <u>0.615±0.012</u> | <u>0.814±0.002</u>  | <b>18.19±0.18</b> |
| TIDER                           | 0.486±0.006         | 0.659±0.009        | 0.833±0.005         | 18.94±0.21        |
| SSD-TS(Ours)                    | <b>0.339±0.0002</b> | <b>0.509±0.007</b> | <b>0.623±0.0001</b> | 18.66±0.26        |

At 70% missing rate, our method ranks second, with only an 11.1% difference from the best-performing CSDI. This demonstrates the strong modeling capability of our method on datasets with high missing rates under random missing scenario.

Table 3 presents the RMSE performance of our method on two real-world datasets with missing data. We observe that on the Physionet dataset, our method achieves the best results across three different missing rates (10%, 50%, and 90%), with improvements of 22.6% (at 10% missing rate), 17.2% (at 50% missing rate), and 23.5% (at 90% missing rate). Additionally, on the AQI dataset, our method ranks second. These results demonstrate that our method outperforms other approaches in modeling time series data on real-world datasets with missing values.

Table 4 presents the CRPS performance of our method on two real-world datasets with missing data. Compared to other probabilistic time series methods, our approach achieves the lowest CRPS scores in three out of the four tasks (10% and 50% on PhysioNet, and AQI). This indicates that, relative to other probabilistic time series modeling methods, our approach demonstrates the most accurate modeling capability of the distribution of unknown points within the sequence.

In addition, we also compare the performance of diffusion models with different backbones. Existing imputation methods mainly cover two kinds of backbones: transformer-based [75] backbones and state-space model-based [22] backbones. The core architecture

**Table 4: CRPS results on Physionet and Air quality dataset. The mean and standard error are obtained by 3 runs and the best result are in bold while the second best results are underlined.**

|                                 | Physionet           |                     |                    | AQI                 |
|---------------------------------|---------------------|---------------------|--------------------|---------------------|
|                                 | 10% missing         | 50% missing         | 90% missing        |                     |
| GP-VAE                          | 0.582±0.003         | 0.796±0.004         | 0.998±0.001        | 0.402±0.009         |
| V-RIN                           | 0.814±0.004         | 0.845±0.002         | 0.932±0.001        | 0.534±0.013         |
| CSDI                            | 0.242±0.001         | 0.336±0.002         | 0.528±0.003        | 0.108±0.001         |
| CSBI                            | 0.247±0.003         | 0.332±0.003         | 0.527±0.006        | 0.110±0.002         |
| SSSD                            | 0.233±0.001         | 0.331±0.002         | <u>0.522±0.002</u> | 0.107±0.001         |
| TS-Diff                         | 0.249±0.002         | 0.348±0.004         | 0.541±0.006        | 0.118±0.003         |
| D <sup>3</sup> M(Constant-Sqrt) | <u>0.223±0.001</u>  | <u>0.327±0.003</u>  | <b>0.520±0.001</b> | <u>0.106±0.002</u>  |
| SSD-TS(Ours)                    | <b>0.164±0.0004</b> | <b>0.244±0.0001</b> | 0.533±0.0004       | <b>0.096±0.0002</b> |

of CSDI is a spatial-temporal transformer, which first embeds the input tokens and captures the inter- and intra- channel dependency with two transformer encoders. D<sup>3</sup>M adopts a similar attention mechanism as the backbone, with the addition of gating mechanisms and an exponential moving average (EMA) module. SSSD modifies the backbone of CSDI by replacing the spatial-temporal transformer with the S4 structure proposed in [22], which is also utilized in TS-Diff.

As stated in Sec.3, our method employs a modern state space model, Mamba [12] as the backbone. Similar to other methods, we also adopt special designs to capture inter- and intra- channel dependencies. The results in Table.3 and 4 show that our method outperforms those with transformer and S4-based core architectures. This demonstrates: 1) the effectiveness of Mamba structure as the backbone for diffusion models and 2) the validity of the designs we adopted in addressing the problem of time series imputation.

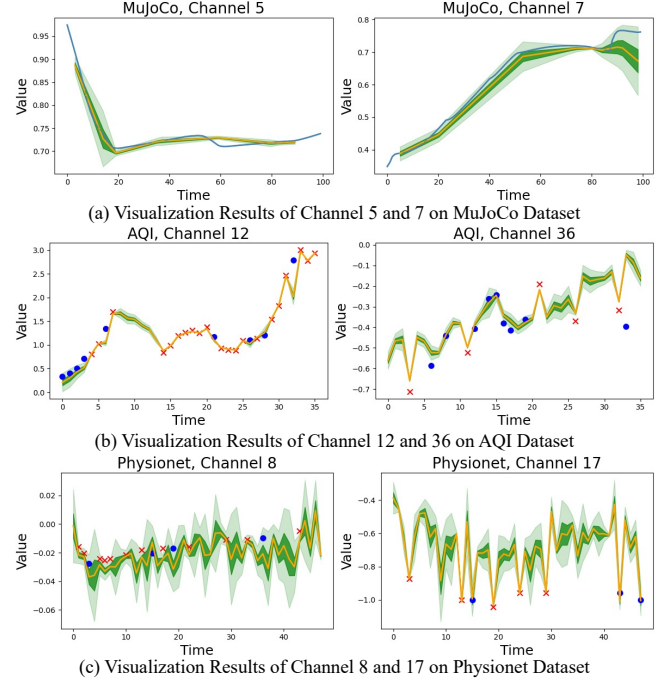
### 4.3 Visualization Results

Fig.4 shows the visualization results on the MuJoCo dataset, AQI dataset and Physionet dataset. We can see that almost all ground truth values for the points to be imputed fall within the 95% confidence interval, and most of the ground truth values are within the 50% confidence interval, which demonstrates the effectiveness of our method.

### 4.4 Ablation Studies

**Effectiveness of Proposed Modules.** To validate the effectiveness of the proposed module, we conduct ablation experiments on the following aspects: 1) the bidirectional modeling, 2) the temporal attention mechanism, 3) the inter-channel multivariate dependencies. We also replace the CMB block with channel attention module implemented using [27] to validate the effectiveness of CMB block. All experiments are conducted on the MuJoCo dataset with the missing ratio 90%.

The results are shown in Table.5. It can be observed that the module equipped with BAM and CMB block performs the best, significantly outperforming the results of removing any one of these components across all four metrics. The temporal attention module has the largest impact on the model, and its removal leads to a significant performance drop. Similarly, removing the CMB module also results in a notable degradation in performance, meanwhile, modifying the CMB block to Channel Attention in [27] also leads to



**Figure 4: Visualization results of probabilistic time series imputation results on MuJoCo, AQI and Physionet dataset. The orange solid line represents the mean of the imputation results, while the dark green represents the 50% confidence interval, and the light green solid line represents the 95% confidence interval. In Figure a, the blue solid line represents the target of imputation (i.e., the ground truth). In Figures b and c, the blue dots represent the ground truth of the missing points, and the red crosses represent the observed values.**

a performance drop, which proves that state space models are also effective for modeling inter-channel dependencies. On the other hand, adjusting the BAM module to its unidirectional form also causes some degree of performance decrease, which means the bidirectional modeling manner is capable of modeling more intra-channel dependency. This fully demonstrates the effectiveness of all our proposed blocks and the capability of linear state space models as backbones for time series diffusion models.

**Time Efficiency of Mamba Backbones.** We compare the inference time of our model with a variant using the transformer backbone. The results is shown in Table.6. We can see that our model achieves faster sampling, less memory cost comparing with the model with transformer backbones, which demonstrates the efficiency of our model and our proposed backbone.

### 4.5 Parameter Sensitivity Analysis

We conduct parameter sensitivity experiment and the results are shown in Table.7. There are three hyperparameters with different dimensions: sequence, residual connection, and input projection dimension, which are set to be equal in our experiments. We test different results for  $C = 32, 64, 128$ . The experimental results show

**Table 5: Experimental Results of Ablation Study**

| Time Modeling | Temporal Attention | Inter-Channel Dependency | MSE                   | MAE                   | MRE                 | RMSE                  |
|---------------|--------------------|--------------------------|-----------------------|-----------------------|---------------------|-----------------------|
| Bidirectional | Yes                | Yes                      | <b>5.46e-4±1.6e-5</b> | <b>1.17e-2±7.4e-5</b> | <b>1.21±7.5e-3%</b> | <b>2.33e-2±3.1e-4</b> |
| Forward       | Yes                | Yes                      | 7.19e-4±2.0e-5        | 1.26e-2±2.1e-4        | 1.29±2.2e-2%        | 2.67e-2±2.9e-4        |
| Forward       | Yes                | No                       | 7.48e-4±9.5e-5        | <u>1.23e-2±3.5e-4</u> | <u>1.23±3.5e-2%</u> | 2.71e-2±1.5e-3        |
| Backward      | Yes                | Yes                      | 7.24e-4±7.3e-5        | 1.30e-2±4.2e-4        | 1.30±4.2e-2%        | 2.69e-2±1.3e-3        |
| Backward      | Yes                | No                       | 8.39e-4±6.1e-5        | 1.46e-2±3.8e-4        | 1.46±3.8e-2%        | 2.89e-2±1.0e-3        |
| Bidirectional | Yes                | No                       | 8.85e-4±2.8e-5        | 1.40e-2±3.1e-4        | 1.44±3.4e-2%        | 2.97e-2±4.8e-4        |
| Bidirectional | No                 | Yes                      | 9.66e-4±9.5e-5        | 1.53e-2±3.3e-4        | 1.57±3.5e-2%        | 3.09e-2±1.3e-3        |
| Bidirectional | Yes                | Channel Attention        | 7.43e-4±4.0e-5        | 1.31e-2±5.5e-5        | 1.35±5.6e-3%        | 2.71e-2±5.6e-5        |

**Table 6: Inference time comparison with transformer backbones**

| Model                    | Transformer-backbone | SSD-TS(128) |
|--------------------------|----------------------|-------------|
| #Parameters(M)           | 9.208                | 87.57       |
| Inference time/sample(s) | 1.5126               | 0.93        |
| GPU Memory(MB)           | 11250                | 4536        |

that as  $C$  increases, all metrics significantly decrease. Additionally, since  $C = 256$  exceeds one single GPU memory capacity, we choose  $C = 128$  in our experiments to balance between metrics and computational cost.

**Table 7: Parameter Sensitivity Results**

| #Channel | MAE    | MSE     | RMSE   |
|----------|--------|---------|--------|
| 32       | 0.0482 | 0.0066  | 0.0809 |
| 64       | 0.0147 | 0.00075 | 0.0273 |
| 128      | 0.0135 | 0.00065 | 0.0254 |

## 5 Related Work

### 5.1 Time Series Imputation

The objective of time series imputation is to recover the missing data points leveraging the observed values and intrinsic dependencies within the series. Statistical time series methods, such as autoregressive (AR) models [18, 45] and autoregressive integrated moving average (ARIMA) models [51, 52], are capable of imputing missing values; however, they often lack the flexibility required to accurately model the complex characteristics of time series data, such as trends and seasonality. Deep learning methods have also found extensive application in imputation tasks. From the perspective of network architectures, various structures including TCN [35], LSTM [25], Multi-Layer Perceptron (MLP), and Transformer [75] have been applied to time series imputation, such as [5, 15, 57, 86, 87, 89]. State space models have also been applied to time series imputation tasks, such as [14, 26]. Recently, methods based on generative models [1, 72, 83] have also achieved great success in probabilistic time series imputation.

### 5.2 Generative Models

Generative models are probabilistic models that learn the original data distribution and generate new samples by sampling from the

learned distribution. The goal is to learn the joint probability distribution  $p(x)$  of the training data or the posterior distribution  $p(x|y)$  and generate new samples that resemble the original data. Traditional generative models include Hidden Markov Models [28, 36], Bayesian models [16, 34] and Gaussian process [10, 91]. Researchers have also proposed many deep learning-based generative models, such as Variational Autoencoders [32], Generative Adversarial Networks (GANs) [19], Normalizing Flow-based Models [33], and Denoising Diffusion Probabilistic Models [24].

Since the goal of generative models is to model the distribution of unknown points and generate target points by sampling from the learned distribution, they naturally possess the ability to estimate the uncertainty of unknown points. This makes generative models particularly well-suited for probabilistic time series modeling. Therefore, many methods have been proposed which adopts generative models for probabilistic time series modeling, including VAE-based methods [13, 37, 77], GAN-based methods [41, 85], Normalizing-flow based methods [11, 64] and diffusion-based methods [63, 69]. State-space models have proved their effectiveness as diffusion backbones [1, 83].

## 6 Conclusion

In this paper, we propose SSD-TS, which explores the potential of linear state-space model (Mamba) as the backbone structure for diffusion models. Based on the requirements of time series modeling task and the characteristics of time series data, we propose a denoising module for diffusion models with Mamba backbone. The proposed module enables bidirectional structural modeling of sequences and effectively leverages both inter-channel and intra-channel dependencies. Experimental results show that, compared to models with structured state-space models, transformers, and convolutional backbones, our method achieves superior performance in time series imputation task. This demonstrates the capability of the linear state-space Mamba model as a backbone for diffusion models. Moreover, our method achieves favorable results in terms of different deterministic and probabilistic metrics, further validating the effectiveness of our design for time series imputation task.

## Acknowledgments

This work was partially supported by National Natural Science Foundation of China (62472174). Jilin Hu is the corresponding author of the work. We gratefully acknowledge the support of the ECNU Multifunctional Platform for Innovation (001) for providing computational resources.

## References

- [1] Juan Miguel Lopez Alcaraz and Nils Strodthoff. 2023. Diffusion-based Time Series Imputation and Forecasting with Structured State Space Models. *Trans. Mach. Learn. Res.* 2023 (2023).
- [2] Lu Bai, Lixin Cui, Yue Wang, Yuhang Jiao, and Edwin R. Hancock. 2020. A Quantum-inspired Entropic Kernel for Multiple Financial Time Series Analysis. 4453–4460 pages.
- [3] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. 2018. BRITS: Bidirectional Recurrent Imputation for Time Series. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*. 6776–6786.
- [4] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8, 1 (2018), 6085.
- [5] Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. 2024. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. *arXiv preprint arXiv:2402.05956* (2024).
- [6] Yu Chen, Wei Deng, Shikai Fang, Fengpei Li, Nicole Tianjiao Yang, Yikai Zhang, Kashif Rasul, Shandian Zhe, Anderson Schneider, and Yuriy Nevmyvaka. 2023. Provably Convergent Schrödinger Bridge with Applications to Probabilistic Time Series Imputation. 4485–4513 pages.
- [7] Tae-Min Choi, Ji-Su Kang, and Jong-Hwan Kim. 2023. RDIS: Random Drop Imputation With Self-Training for Incomplete Time Series Data. *IEEE Access* 11 (2023), 100720–100728.
- [8] Andrea Cini, Ivan Marisca, and Cesare Alippi. 2022. Filling the G<sub>ap</sub>s: Multivariate Time Series Imputation by Graph Neural Networks. In *The Tenth International Conference on Learning Representations, ICLR 2022*. OpenReview.net.
- [9] Razvan-Gabriel Cirstea, Chenjuan Guo, Bin Yang, Tung Kieu, Xuanyi Dong, and Shirui Pan. 2022. Triformer: Triangular, Variable-Specific Attentions for Long Sequence Multivariate Time Series Forecasting. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022*. 1994–2001.
- [10] Giorgio Corani, Alessio Benavoli, and Marco Zaffalon. 2021. Time Series Forecasting with Gaussian Processes Needs Priors. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track - European Conference, ECML PKDD 2021 (Lecture Notes in Computer Science, Vol. 12978)*. Springer, 103–117.
- [11] Enyan Dai and Jie Chen. 2022. Graph-Augmented Normalizing Flows for Anomaly Detection of Multiple Time Series. In *The Tenth International Conference on Learning Representations, ICLR 2022*.
- [12] Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. In *Forty-first International Conference on Machine Learning, ICML 2024*.
- [13] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. 2021. TimeVAE: A Variational Auto-Encoder for Multivariate Time Series Generation. *CoRR abs/2111.08095* (2021).
- [14] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2017. Density estimation using Real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*. OpenReview.net.
- [15] Wenjie Du, David Côté, and Yan Liu. 2023. SAITS: Self-attention-based imputation for time series. *Expert Syst. Appl.* 219 (2023), 119619.
- [16] Shikai Fang, Qingsong Wen, Yingtao Luo, Shandian Zhe, and Liang Sun. 2024. BayOTIDE: Bayesian Online Multivariate Time Series Imputation with Functional Decomposition. In *Forty-first International Conference on Machine Learning, ICML 2024*.
- [17] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. 2020. GP-VAE: Deep Probabilistic Time Series Imputation. 1651–1661 pages.
- [18] Wayne A Fuller and David P Hasza. 1981. Properties of predictors for autoregressive time series. *J. Amer. Statist. Assoc.* 76, 373 (1981), 155–161.
- [19] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*. 2672–2680.
- [20] Albert Gu and Tri Dao. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces.
- [21] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. 2020. HiPPo: Recurrent Memory with Optimal Polynomial Projections. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.
- [22] Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently Modeling Long Sequences with Structured State Spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- [23] Chenjuan Guo, Christian S. Jensen, and Bin Yang. 2014. Towards Total Traffic Awareness. *SIGMOD Record* 43, 3 (2014), 18–23.
- [24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.
- [25] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (1997), 1735–1780.
- [26] Jiaxi Hu, Disen Lan, Ziyu Zhou, Qingsong Wen, and Yuxuan Liang. 2024. Time-SSM: Simplifying and Unifying State Space Models for Time Series Forecasting. *CoRR abs/2405.16312* (2024).
- [27] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-Excitation Networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*. Computer Vision Foundation / IEEE Computer Society, 7132–7141.
- [28] Yexin Huang. 2023. Pyramidal Hidden Markov Model For Multivariate Time Series Forecasting. *CoRR abs/2310.14341* (2023).
- [29] Zahra Karevan and Johan A. K. Suykens. 2020. Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks* 125 (2020), 1–9.
- [30] Seunghyun Kim, Hyunsu Kim, Eunghu Yun, Hwangrae Lee, Jaehun Lee, and Juho Lee. 2023. Probabilistic Imputation for Time-series Classification with Missing Data. 16654–16667 pages.
- [31] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015*.
- [32] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014*.
- [33] Ivan Kobyzev, Simon J. D. Prince, and Marcus A. Brubaker. 2021. Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* 43, 11 (2021), 3964–3979.
- [34] Ozan Kocadagli and Baris Asikgil. 2014. Nonlinear time series forecasting with Bayesian neural networks. *Expert Syst. Appl.* 41, 15 (2014), 6596–6610.
- [35] Colin Lea, Michael D. Flynn, René Vidal, Austin Reiter, and Gregory D. Hager. 2017. Temporal Convolutional Networks for Action Segmentation and Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. IEEE Computer Society, 1003–1012.
- [36] Jing Li, Botong Wu, Xinwei Sun, and Yizhou Wang. 2021. Causal Hidden Markov Model for Time Series Disease Forecasting. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021*. Computer Vision Foundation / IEEE, 12105–12114.
- [37] Longyuan Li, Junchi Yan, Haiyang Wang, and Yaohui Jin. 2021. Anomaly Detection of Time Series With Smoothness-Inducing Sequential Variational Auto-Encoder. *IEEE Trans. Neural Networks Learn. Syst.* 32, 3 (2021), 1177–1191.
- [38] Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, and Bin Yang. 2025. TSMF-Bench: Comprehensive and Unified Benchmarking of Foundation Models for Time Series Forecasting. In *SIGKDD*.
- [39] Shuai Liu, Xiucheng Li, Gao Cong, Yile Chen, and Yue Jiang. 2023. Multivariate Time-series Imputation with Disentangled Temporal Representations. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net.
- [40] Sicong Liu, Yang Shu, Chenjuan Guo, and Bin Yang. 2025. Learning Generalizable Skills from Offline Multi-Task Data for Multi-Agent Cooperation. In *The Thirteenth International Conference on Learning Representations, ICLR*. OpenReview.net.
- [41] Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, and Xiaojie Yuan. 2018. Multivariate Time Series Imputation with Generative Adversarial Networks. 1603–1614 pages.
- [42] Haoyu Ma, Yushu Chen, Wenlai Zhao, Jinzhe Yang, Yingsheng Ji, Xinghua Xu, Xiaozhuo Liu, Hao Jing, Shengzhuo Liu, and Guangwen Yang. 2024. A Mamba Foundation Model for Time Series Forecasting. *CoRR abs/2411.02941* (2024).
- [43] Ivan Marisca, Andrea Cini, and Cesare Alippi. 2022. Learning to Reconstruct Missing Data from Spatiotemporal Graphs with Sparse Observations. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.).
- [44] Amy McGovern, Derek H. Rosendahl, Rodger A. Brown, and Kelvin Droegemeier. 2011. Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction. *Data Min. Knowl. Discov.* 22, 1-2 (2011), 232–258.
- [45] Christopher Meek, David Maxwell Chickering, and David Heckerman. 2002. Autoregressive tree models for time-series analysis. In *Proceedings of the 2002 SIAM International Conference on Data Mining*. SIAM, 229–244.
- [46] Hao Miao, Yan Zhao, Chenjuan Guo, Bin Yang, Kai Zheng, Feiteng Huang, Jiantong Xie, and Christian S. Jensen. 2024. A Unified Replay-Based Continuous Learning Framework for Spatio-Temporal Prediction on Streaming Data. In *ICDE*. 1050–1062.
- [47] Xiaoye Miao, Yangyang Wu, Jun Wang, Yunjun Gao, Xudong Mao, and Jianwei Yin. 2021. Generative Semi-supervised Learning for Multivariate Time Series Imputation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*. AAAI Press, 8983–8991.
- [48] Mohammad Amin Morid, Olivia R. Liu Sheng, and Joseph Dunbar. 2023. Time Series Prediction Using Deep Learning Methods in Healthcare. 2:1–2:29 pages.
- [49] James Morrill, Cristopher Salvi, Patrick Kidger, and James Foster. 2021. Neural Rough Differential Equations for Long Time Series. In *Proceedings of the 38th*

- International Conference on Machine Learning, ICML 2021 (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 7829–7838.
- [50] Ahmad Wisnu Mulyadi, Eunji Jun, and Heung-II Suk. 2022. Uncertainty-Aware Variational-Recurrent Imputation Network for Clinical Time Series. *IEEE Trans. Cybern.* 52, 9 (2022), 9684–9694.
  - [51] Brian K Nelson. 1998. Time series analysis using autoregressive integrated moving average (ARIMA) models. *Academic emergency medicine* 5, 7 (1998), 739–744.
  - [52] Paul Newbold. 1983. ARIMA model building and the time series analysis approach to forecasting. *Journal of forecasting* 2, 1 (1983), 23–35.
  - [53] Tong Nie, Guoyang Qin, Wei Ma, Yuewen Mei, and Jian Sun. 2024. ImputeFormer: Low Rankness-Induced Transformers for Generalizable Spatiotemporal Imputation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024*. ACM, 2260–2271.
  - [54] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023*.
  - [55] Biao Ouyang, Yingying Zhang, Hanyin Cheng, Yang Shu, Chenjuan Guo, Bin Yang, Qingsong Wen, Lunting Fan, and Christian S. Jensen. 2025. RCRank: Multimodal Ranking of Root Causes of Slow Queries in Cloud Database Systems. *CoRR abs/2503.04252* (2025).
  - [56] Patrick Asante Owusu, Etienne Gael Tajeuna, Jean-Marc Patenaude, Armelle Brun, and Shengrui Wang. 2023. Rethinking Temporal Dependencies in Multiple Time Series: A Use Case in Financial Data. 1247–1252 pages.
  - [57] Jangho Park, Juliane Müller, Bhavna Arora, Boris Faybishenko, Gilberto Zonta Pastorello, Charuleka Varadharajan, Reetik Sahu, and Deborah A. Agarwal. 2023. Long-term missing value imputation for time series data using deep neural networks. *Neural Comput. Appl.* 35, 12 (2023), 9071–9091.
  - [58] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library.
  - [59] Onur Poyraz and Pekka Marttinen. 2023. Mixture of Coupled HMMs for Robust Modeling of Multivariate Healthcare Time Series. 461–479 pages.
  - [60] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. 2024. TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods. In *Proc. VLDB Endow.* 2363–2377.
  - [61] Xiangfei Qiu, Zhe Li, Wanghui Qiu, Shiyuan Hu, Lekui Zhou, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Aoying Zhou, Zhenli Sheng, Jilin Hu, Christian S. Jensen, and Bin Yang. 2025. TAB: Unified Benchmarking of Time Series Anomaly Detection Methods. In *Proc. VLDB Endow.*
  - [62] Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. 2025. DUET: Dual Clustering Enhanced Multivariate Time Series Forecasting. In *SIGKDD*. 1185–1196.
  - [63] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. 2021. Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021 (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 8857–8868.
  - [64] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs M. Bergmann, and Roland Vollgraf. 2021. Multivariate Probabilistic Time Series Forecasting via Conditioned Normalizing Flows. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net.
  - [65] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI (Lecture Notes in Computer Science, Vol. 9351)*. 234–241.
  - [66] Yulia Rubanova, Tian Qi Chen, and David Duvenaud. 2019. Latent Ordinary Differential Equations for Irregularly-Sampled Time Series. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*. 5321–5331.
  - [67] Patrick Seifner, Kostadin Cvejovski, Antonia Körner, and Ramsés J. Sánchez. 2025. Zero-shot Imputation with Foundation Inference Models for Dynamical Systems. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*. OpenReview.net.
  - [68] Siyuan Shan, Yang Li, and Junier B Oliva. 2023. Nrtsi: Non-recurrent time series imputation. 5 pages.
  - [69] Lifeng Shen, Weiyu Chen, and James T. Kwok. 2024. Multi-Resolution Diffusion Models for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna*.
  - [70] Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. 2012. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 computing in cardiology*. IEEE, 245–248.
  - [71] Qiuling Suo, Weida Zhong, Guangxu Xun, Jianhui Sun, Changyou Chen, and Aidong Zhang. 2020. GLIMA: Global and local time series imputation with multi-directional attention learning. 798–807 pages.
  - [72] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. 2021. CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation. 24804–24816 pages.
  - [73] Jindong Tian, Yuxuan Liang, Ronghui Xu, Peng Chen, Chenjuan Guo, Aoying Zhou, Lujia Pan, Zhongwen Rao, and Bin Yang. 2025. Air quality prediction with physics-guided dual neural odes in open systems. In *The Thirteenth International Conference on Learning Representations*.
  - [74] Maria Tzelepi, Paraskevi Nousi, and Anastasios Tefas. 2023. Improving Electric Load Demand Forecasting with Anchor-Based Forecasting Method. 5 pages.
  - [75] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*. 5998–6008.
  - [76] Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. 2025. Is Mamba effective for time series forecasting? *Neurocomputing* 619 (2025), 129178.
  - [77] Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Chenjuan Guo, and Bin Yang. 2025. K<sup>2</sup>VAE: A Koopman-Kalman Enhanced Variational AutoEncoder for Probabilistic Time Series Forecasting. In *ICML*.
  - [78] Xingjian Wu, Xiangfei Qiu, Zhengyu Li, Yihang Wang, Jilin Hu, Chenjuan Guo, Hui Xiong, and Bin Yang. 2025. CATCH: Channel-Aware multivariate Time Series Anomaly Detection via Frequency Patching. In *ICLR*.
  - [79] Xinle Wu, Dalin Zhang, Miao Zhang, Chenjuan Guo, Bin Yang, and Christian S. Jensen. 2023. AutoCTS+: Joint Neural Architecture and Hyperparameter Search for Correlated Time Series Forecasting. *Proc. ACM Manag. Data* 1, 1 (2023), 97:1–97:26.
  - [80] Sheng Xiang, Dawei Cheng, Chencheng Shang, Ying Zhang, and Yuqi Liang. 2022. Temporal and Heterogeneous Graph Neural Network for Financial Time Series Prediction. 3584–3593 pages.
  - [81] Ronghui Xu, Hanyin Cheng, Chenjuan Guo, Hongfan Gao, Jilin Hu, Sean Bin Yang, and Bin Yang. 2025. MM-Path: Multi-modal, Multi-granularity Path Representation Learning. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*. 1703–1714.
  - [82] Ronghui Xu, Weiming Huang, Jun Zhao, Meng Chen, and Liqiang Nie. 2023. A Spatial and Adversarial Representation Learning Approach for Land Use Classification with POIs. *ACM Transactions on Intelligent Systems and Technology* 14, 6 (2023), 1–25.
  - [83] Tijin Yan, Hengheng Gong, Yongping He, Yufeng Zhan, and Yuanqing Xia. 2024. Probabilistic Time Series Modeling with Decomposable Denoising Diffusion Model. In *Forty-first International Conference on Machine Learning, ICML 2024*. OpenReview.net.
  - [84] Xiwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. 2016. ST-MVL: Filling Missing Values in Geo-Sensory Time Series Data. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*. IJCAI/AAAI Press, 2704–2710.
  - [85] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series Generative Adversarial Networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*. 5509–5519.
  - [86] Hongwu Yuan, Guoming Xu, Zijian Yao, Ji Jia, and Yiwen Zhang. 2018. Imputation of Missing Data in Time Series for Air Pollutants Using Long Short-Term Memory Recurrent Neural Networks. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, UbiComp/ISWC 2018 Adjunct*. ACM, 1293–1300.
  - [87] A. Yarkin Yildiz, Emirhan Koç, and Aykut Koç. 2022. Multivariate Time Series Imputation With Transformers. *IEEE Signal Processing Letters* 29 (2022), 2517–2521.
  - [88] Biao Zhang and Rico Sennrich. 2019. Root Mean Square Layer Normalization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*. 12360–12371.
  - [89] Kai Zhang, Chao Li, and Qinmin Yang. 2023. TriD-MAE: A Generic Pre-trained Model for Multivariate Time Series with Missing Values. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023*. ACM, 3164–3173.
  - [90] Tengxue Zhang, Yang Shu, Xinyang Chen, Yifei Long, Chenjuan Guo, and Bin Yang. 2025. Assessing Pre-Trained Models for Transfer Learning Through Distribution of Spectral Components. In *AAAI-25*. 22560–22568.
  - [91] Zhendong Zhang, Chao Wang, Xiaosheng Peng, Hui Qin, Hao Lv, Jialong Fu, and Hongyu Wang. 2021. Solar Radiation Intensity Probabilistic Forecasting Based on K-Means Time Series Clustering and Gaussian Process Regression. *IEEE Access* 9 (2021), 89079–89092.
  - [92] Kai Zhao, Chenjuan Guo, Yunyao Cheng, Peng Han, Miao Zhang, and Bin Yang. 2023. Multiple Time Series Forecasting with Dynamic Graph Modeling. *Proc. VLDB Endow.* 17, 4 (2023), 753–765.
  - [93] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. 11106–11115 pages.

## 7 Appendix

### 7.1 Experiment Details

**7.1.1 Dataset Descriptions.** In this part, we give a brief introduction about the datasets in our experiments.

**Air quality:** The air quality dataset contains PM2.5 data from 36 monitor stations in Beijing, which is sampled hourly for 12 months. There are 13.3% of missing values with a non-random missing pattern. The air quality dataset contains artificial ground truth with structured missing pattern.

**Healthcare (Physionet):** The healthcare dataset contains 4000 irregularly-sampled clinical time series made up of 35 variables (such as Albumin and heart-rate) for 48 hours collected from ICU. To be consistent with previous studies, the dataset is processed hourly to get 48 timesteps and the processed dataset contains near 80% missing values without ground truth. For evaluation, we randomly choose 10/50/90% of the observed values as the ground truth of test dataset.

**MuJoCo:** The MuJoCo dataset [66] collects a total of 10,000 simulations of the "Hopper" model from the DeepMind Control Suite and MuJoCo simulator. The position of the body in 2D space is uniformly sampled from the interval  $[0, 0.5]$ . The relative position of the limbs is sampled from the range  $[-2, 2]$ , and initial velocities are sampled from the interval  $[-5, 5]$ . In all, there are 10000 sequences of 100 regularly sampled time points with a feature dimension of 14 and a random split of 80/20 is done for training and testing. We follow the same preprocessing as in [68] for fair comparison.

**7.1.2 Hyperparameters.** Table 8 lists the hyperparameters in SSD-TS.

**Table 8: Hyperparameters in SSD-TS**

|   | SSD-TS |
|---|--------|
| Sequence dim ( $C$ in Fig.2)                                      | 128    |
| Residual channels ( $K$ in Fig.2)                                 | 128    |
| Num channels (dim of input projections before $\epsilon_\theta$ ) | 128    |
| Diffusion embedding dim   | 128    |
| Training iteration  | 150k   |
| Num of conditional SMM blocks                                     | 1      |
| Num of input SMM blocks   | 1      |
| Num of sequential SMM blocks                                      | 1      |

**7.1.3 Time Series Forecasting Results.** Our model is applicable to time series forecasting tasks and we test it on the ETm1 [93] dataset with forecasting length = 24 and 96, the results are in Table.9. Our model ranks 1st in 3 of 4 metrics and 2nd on MSE of forecasting length 24, while also performs best among imputation models.

**7.1.4 Block Missing Results.** To evaluate the performance of our models on block missing scenario, we trained our models on the PTB-XL dataset (in [1]) and the result of 20% block missing is presented in Table.10.

**7.1.5 Additional Time Series Imputation Results.** We also compare our SSD-TS with FIM [67](on Physionet), ImputeFormer [53](on AQI) and Bayotide [16](on Solar dataset). As the source code of these

**Table 9: Time series forecasting results on ETm1 dataset**

| Forecasting Length | 24           |              | 96           |              |
|--------------------|--------------|--------------|--------------|--------------|
| Model              | MAE          | MSE          | MAE          | MSE          |
| LSTNet             | 1.170        | 1.968        | 1.542        | 2.762        |
| LSTMa              | 0.629        | 0.621        | 0.913        | 1.339        |
| Reformer           | 0.607        | 0.724        | 0.945        | 1.433        |
| LogTrans           | 0.412        | 0.419        | 0.792        | 0.768        |
| Informer           | 0.369        | <b>0.323</b> | 0.614        | 0.678        |
| CSDI               | 0.370        | 0.354        | 0.756        | 1.468        |
| Autoformer         | 0.403        | 0.383        | <u>0.463</u> | <u>0.481</u> |
| SSSD               | <u>0.361</u> | 0.351        | 0.547        | 0.538        |
| SSD-TS(Ours)       | <b>0.282</b> | <u>0.331</u> | <b>0.391</b> | <b>0.421</b> |

**Table 10: Block missing results on PTB-XL dataset.**

| Model    | MAE           | RMSE          |
|----------|---------------|---------------|
| LAMC     | 0.0840        | 0.1171        |
| CSDI     | 0.1054        | 0.2254        |
| DiffWave | 0.0451        | 0.1378        |
| SSSD     | <u>0.0324</u> | <u>0.0832</u> |
| SSD-TS   | <b>0.022</b>  | <b>0.059</b>  |

baselines are not provided, all the results are collected from their corresponding papers. For FIM model, we report its best results.

**Table 11: Comparison with FIM on physionet dataset**

| Models | BRITS | SAITS | CSDI         | FIM- $l(o.n. = 4)$ | SSD-TS       |
|--------|-------|-------|--------------|--------------------|--------------|
| MAE    | 0.297 | 0.257 | <u>0.252</u> | 0.402              | <b>0.217</b> |

**Table 12: Comparison with Imputeformer on AQI dataset**

| Models | BRITS | SAITS | TIDER | GRIN  | SPIN  | ImputeFormer | CSDI        | SSD-TS      |
|--------|-------|-------|-------|-------|-------|--------------|-------------|-------------|
| MAE    | 14.74 | 19.79 | 32.85 | 12.08 | 11.89 | 11.58        | <u>9.60</u> | <b>6.75</b> |

**Table 13: Comparison with Bayotide on Solar dataset (50% missing rate)**

| Models | BRITS | NAOMI | SAITS | TIDER        | CSDI  | CSBI  | BayoTIDE     | SSD-TS       |
|--------|-------|-------|-------|--------------|-------|-------|--------------|--------------|
| RMSE   | 2.842 | 2.918 | 2.791 | <u>1.679</u> | 2.276 | 2.097 | 1.699        | <b>0.868</b> |
| MAE    | 1.985 | 2.112 | 1.827 | 0.838        | 0.804 | 1.033 | <u>0.734</u> | <b>0.617</b> |

**Table 14: Comparison with Bayotide on Solar dataset (30% missing rate)**

| Models | BRITS | NAOMI | SAITS | TIDER | CSDI  | CSBI  | BayoTIDE     | SSD-TS       |
|--------|-------|-------|-------|-------|-------|-------|--------------|--------------|
| RMSE   | 2.617 | 2.702 | 2.359 | 1.676 | 2.132 | 1.987 | <u>1.621</u> | <b>0.987</b> |
| MAE    | 1.861 | 2.003 | 1.575 | 0.874 | 1.045 | 0.926 | <u>0.709</u> | <b>0.648</b> |