# SPECSTG: A FAST SPECTRAL DIFFUSION FRAMEWORK FOR PROBABILISTIC SPATIO-TEMPORAL TRAFFIC FORECASTING

**Lequan Lin** *
University of Sydney, Australia

**Dai Shi** *
University of Sydney, Australia

**Andi Han**
Riken AIP, Japan

**Junbin Gao**
University of Sydney, Australia

## ABSTRACT

Traffic forecasting, a crucial application of spatio-temporal graph (STG) learning, has traditionally relied on deterministic models for accurate point estimations. Yet, these models fall short of quantifying future uncertainties. Recently, many probabilistic methods, especially variants of diffusion models, have been proposed to fill this gap. However, existing diffusion methods typically deal with individual sensors separately when generating future time series, resulting in limited usage of spatial information in the probabilistic learning process. In this work, we propose SpecSTG, a novel spectral diffusion framework, to better leverage spatial dependencies and systematic patterns inherent in traffic data. More specifically, our method generates the Fourier representation of future time series, transforming the learning process into the spectral domain enriched with spatial information. Additionally, our approach incorporates a fast spectral graph convolution designed for Fourier input, alleviating the computational burden associated with existing models. Compared with state-of-the-arts, SpecSTG achieves up to 8% improvements on point estimations and up to 0.78% improvements on quantifying future uncertainties. Furthermore, SpecSTG's training and validation speed is 3.33× of the most efficient existing diffusion method for STG forecasting. The source code for SpecSTG is available at https://github.com/lequanlin/SpecSTG.

## 1 INTRODUCTION

Traffic forecasting on road networks is an essential application domain of spatio-temporal graph (STG) learning (Yuan & Li, 2021; Jin et al., 2023a;b). As a crucial component of the Intelligence Transportation System (ITS), accurate and informative prediction of future traffic dynamics provides essential guidelines to traffic authorities in decision-making (Lana et al., 2018; Boukerche et al., 2020). Since traffic data, such as vehicle speed and traffic flow, are collected from sensors in the continuous space of road networks, they present strong spatio-temporal dependencies, especially at neighbouring locations and time windows (Guo et al., 2019). This naturally leads to their representation as STGs: the traffic network is modelled as a graph, in which nodes are sensors and edges are decided with some criteria such as geographic distances. Hence, temporal records are stored as graph signals, while spatial information is encapsulated in the graph structure. In Figure 1(a), we provide a visualized example of traffic STG.

STG traffic forecasting aims at predicting future values at all sensors based on past time series and spatial connections in the traffic network. This task has traditionally relied on deterministic models such as DCRNN (Li et al., 2018) and GMAN (Zheng et al., 2020) to produce accurate point estimations. Nevertheless, these models may fall short in identifying unexpected variations that lead to consequential change in traffic regulations (Pal et al., 2021; Wen et al., 2023; Hu et al., 2023). This limitation can be overcome by probabilistic methods, which alternatively approximate the distribution of future time series, thus leading to more uncertainty-aware predictions. For example, Figure

*Equal contributions. Corresponding to Lequan Lin lequan.lin@sydney.edu.au.

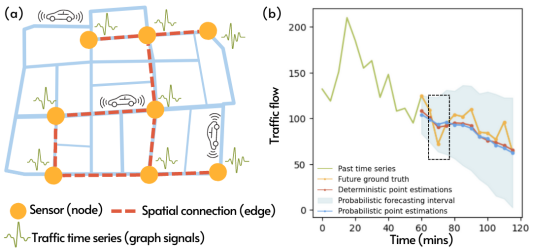


Figure 1: Illustrations: (a) an example of traffic STG; (b) traffic flow forecasting in future 60 minutes with GMAN (deterministic) and SpecSTG (probabilistic) on PEMS04.

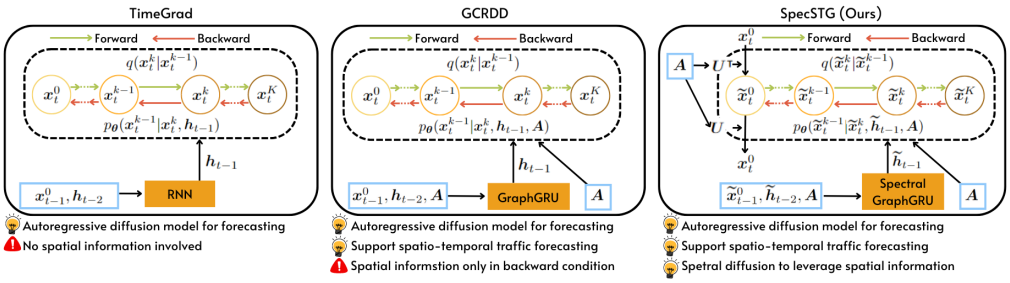


Figure 2: A comparison of SpecSTG and two other existing methods. SpecSTG adopts a spectral diffusion process that generates the graph Fourier representation of future time series, thus better leveraging spatial information in learning and prediction.

1(b) shows the results of deterministic and probabilistic models on the PEMS04 traffic flow forecasting task (Guo et al., 2019). The deterministic model can only provide point estimations (red), while the probabilistic model is capable of generating both point estimations and the forecasting interval (blue) which captures some abrupt fluctuations in traffic flow (black box).

Among all the probabilistic models applicable to STG traffic forecasting, we specifically focus on diffusion models (Yang et al., 2023; Ho et al., 2020; Lin et al., 2023). Classic diffusion models for time series forecasting, such as TimeGrad (Rasul et al., 2021a), generally follow a forward-backward training process: the generative target (i.e., future time series) is first turned into white noise and then recovered by a learnable backward kernel. The backward kernel is a conditional distribution which is similar to traditional diffusion models except for the inclusion of encoded past temporal information in its conditions. For prediction, samples from the distribution of future time series are generated by denoising white noise with the learned backward kernel. Since such methods are incapable of handling spatial information in STGs, diffusion models for STG forecasting, such as DiffSTG (Wen et al., 2023) and GCRDD (Li et al., 2023), incorporate graph structure in the backward kernel condition. More specifically, the backward kernel is usually approximated by a denoising network, which takes graph structure and other conditions as input to predict noises injected in the forward process. More related works are discussed in Appendix A.

Our work considers two limitations of existing diffusion methods for STG forecasting. (**Limitation #1**) Although these methods emphasize on the importance of spatial information in STG forecasting, they only use spatial information in the backward kernel condition. Consequently, the involvement of spatial information in the overall forward-backward diffusion learning process is limited. (**Limitation #2**) The denoising network of existing methods usually relies substantially on graph convolution to encode spatial information. This often introduces a complexity of $\mathcal{O}(N^2)$, quadratic in the number of sensors $N$ in a traffic STG (see Subsection 4.1 for more details). Thus the computational cost is high especially for large traffic networks, leading to slow training and sampling.

In this work, we propose a novel spectral diffusion framework (SpecSTG), which adopts the graph Fourier representation of time series as the generative target. According to the spectral graph the-

ory, the graph Fourier representation is a measurement of variations in graph signals guided by the graph structure (Chung, 1997; Kreuzer et al., 2021). In the context of STGs, we may treat time points as features, thereby the graph Fourier representation can be considered as a new time series of systematic fluctuations enriched by spatial information. Hence, **Limitation #1** is resolved by generating the Fourier representation rather than the original time series, transforming the entire diffusion process into the spectral domain. This effectively leverages the graph structure to construct a more comprehensive diffusion base with additional systematic and spatial patterns. Besides, with no loss of information, the generated data can be converted back to the original domain via the inverse Fourier transform for prediction. **Limitation #2** is mitigated by replacing the graph convolution with a light-complexity alternative, which only works for the Fourier input (details in Subsection 4.1). An overview of SpecSTG can be found in Figure 2. We also provide illustrations of TimeGrad and GCRDD for comparison. The contributions of this paper are three-fold:

(1) To our best knowledge, this is the first work that explores probabilistic STG forecasting on the graph spectral domain.

(2) SpecSTG achieves up to 8% improvements on point estimations and up to 0.78% improvements on generating compatible forecasting intervals.

(3) SpecSTG's training and validation speed is 3.33× of the most efficient existing diffusion method for STG forecasting. Additionally, SpecSTG significantly accelerates the sampling process, particularly for large sample sizes.

## 2 PRELIMINARIES

### 2.1 SPATIO-TEMPORAL GRAPHS (STGS) & STG FORECASTING

STGs can be considered as a multidimensional graph representation of entities in a system with time series as graph signals. In traffic forecasting, we model sensors as nodes and then create edges based on some spatial relationships such as geographic distances. The average traffic records in observation periods are modelled as graph signals. For a traffic network with $N$ sensors, the corresponding STG can be denoted as $\mathcal{G}\{\mathcal{V}, \mathcal{E}, \boldsymbol{A}\}$, where $\mathcal{V}$ is the set of nodes/sensors, $\mathcal{E}$ is the set of edges, and $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix. $\boldsymbol{A}$ is assumed to be undirected and can be either weighted or unweighted. The graph signals are denoted as $\boldsymbol{X}_{\mathcal{G}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_t, ...|\boldsymbol{x}_t \in \mathbb{R}^{N\times D_x}\}$, where $D_x$ is the number of variables. In traffic forecasting, it is common that only one variable such as speed or flow is of interest (Li et al., 2018; Guo et al., 2019), thus we usually have $D_x = 1$.

The objective of STG traffic forecasting is to predict a future time series window $\boldsymbol{X}_f = \{\boldsymbol{x}_{t_0+1}, \boldsymbol{x}_{t_0+2}, ..., \boldsymbol{x}_{t_0+f}\}$ given the past context window $\boldsymbol{X}_c = \{\boldsymbol{x}_{t_0-c+1}, \boldsymbol{x}_{t_0-c+2}, ..., \boldsymbol{x}_{t_0}\}$, where $f$ and $c$ are the length of future and past windows. We denote the combination of past and future time series as $\boldsymbol{X} = \{\boldsymbol{x}_{t_0-c+1}, \boldsymbol{x}_{t_0-c+2}, ..., \boldsymbol{x}_{t_0+f}\}$. Normally, the target distribution of generative models depends on the sampling methods: one-shot methods produce all future predictions together from the distribution $q(\boldsymbol{X}_f|\boldsymbol{X}_c, \boldsymbol{A})$ (Wen et al., 2023; Liu et al., 2023), while autoregressive methods generate samples from $q(\boldsymbol{x}_t|\boldsymbol{X}_{t_0-c+1:t-1}, \boldsymbol{A})$ for $t = t_0+1, t_0+2, ..., t_0+f$ successively, where $\boldsymbol{X}_{t_0-c+1:t-1} = \{\boldsymbol{x}_{t_0-c+1}, ..., \boldsymbol{x}_{t-1}\}$ (Li et al., 2023). Autoregressive methods often capture the sequential information in consecutive time points more closely. However, they are associated with higher time costs because of the non-parallel step-by-step sampling process. Our method, SpecSTG, is formulated within the autoregressive framework but is equipped with a specially designed spectral graph convolution to mitigate computational inefficiency.

### 2.2 DENOISING DIFFUSION PROBABILISTIC MODEL

Denoising diffusion probabilistic models (DDPMs) learn how to generate samples from the target distribution via a pair of forward-backward Markov chains (Ho et al., 2020; Yang et al., 2023). Assuming that $\boldsymbol{x}^0 \sim q(\boldsymbol{x}^0)$ is the original data, for diffusion step $k = 0, 1, ..., K$, the forward chain injects Gaussian noises to $\boldsymbol{x}^k$ until $q(\boldsymbol{x}^K) := \int q(\boldsymbol{x}^K|\boldsymbol{x}^0)q(\boldsymbol{x}^0)\mathrm{d}\boldsymbol{x}^0 \approx \mathcal{N}(\boldsymbol{x}^K; \boldsymbol{0}, \boldsymbol{I})$. As a special property, given a noise schedule $\beta = \{\beta_1, \beta_2, ..., \beta_K\}$, we may directly compute the disturbed data at step $k$ as $\boldsymbol{x}^k = \sqrt{\tilde{\alpha}_k}\boldsymbol{x}^0 + \sqrt{1-\tilde{\alpha}_k}\boldsymbol{\epsilon}$, where $\tilde{\alpha}_k = \prod_{i=1}^{k}(1-\beta_i)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Next, the backward chain denoises from $\boldsymbol{x}^K$ to recover $p_{\boldsymbol{\theta}}(\boldsymbol{x}^0)$ through a probabilistic backward kernel

$p_{\boldsymbol{\theta}}(\boldsymbol{x}^{k-1}|\boldsymbol{x}^k)$, where $\boldsymbol{\theta}$ denotes all learnable parameters. In practice, the backward kernel is usually optimized with a denoising network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ by minimizing the loss function

$$\mathcal{L}_{DDPM}(\boldsymbol{\theta}) = \mathbb{E}_{k,\boldsymbol{x}^0,\boldsymbol{\epsilon}} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}} \left( \boldsymbol{x}^k, k \right) \right\|^2, \tag{1}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ represents the noises injected in the forward diffusion steps.

## 3 STG Fourier Transform

Given a traffic STG $\mathcal{G}\{\mathcal{V}, \mathcal{E}, \boldsymbol{A}\}$ with $N$ sensors, the normalized graph Laplacian is computed as $\boldsymbol{L} = \boldsymbol{I}_N - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{I}_N \in \mathbb{R}^{N\times N}$ is the identity matrix, and $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ is the diagonal degree matrix. We denote the eigen-decomposition of the graph Laplacian as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\intercal$, where $\boldsymbol{U} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{\Lambda} \in \mathbb{R}^{N\times N}$ are the corresponding eigenvector and eigenvalue matrices, respectively. Considering the univariate graph signal $\boldsymbol{X}_{\mathcal{G}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_t, ...|\boldsymbol{x}_t \in \mathbb{R}^N\}$, the Fourier transform for each time point $t$ is given by $\tilde{\boldsymbol{x}}_t = \boldsymbol{U}^\intercal \boldsymbol{x}_t$, known as the Fourier representation of $\boldsymbol{x}_t$ in the spectral domain. The Fourier reconstruction is $\boldsymbol{x}_t = \boldsymbol{U}\tilde{\boldsymbol{x}}_t$. The orthonormal $\boldsymbol{U}$ ensures a lossless reconstruction for the temporal information. We may compute in matrix form for all time points as $\tilde{\boldsymbol{X}} = \boldsymbol{U}^\intercal \boldsymbol{X}$ and $\boldsymbol{X} = \boldsymbol{U}\tilde{\boldsymbol{X}}$. In Appendix B, we also discuss how to naturally extend the method to multivariate traffic STGs with $\boldsymbol{x}_t \in \mathbb{R}^{N\times D_x}$, $D_x \geq 2$.

The graph Fourier transform can be understood as a projection of graph signals onto the spectral domain spanned by the eigenvector basis of graph Laplacian. The operator $\boldsymbol{U}$ brings rich positional information for graph signals and offers a platform to investigate the variations among signals through a global perspective on the graph. In particular, when the input is a traffic STG, the Fourier representation measures how graph signals (time series) fluctuate across the network. This effectively integrates spatial connectivity into time series, leading to a spatial-aware forecasting paradigm.

## 4 The Proposed Method

SpecSTG assumes that the Fourier representation of future time series follows the distribution

$$q(\tilde{\boldsymbol{X}}_f^0 | \tilde{\boldsymbol{X}}_c^0, \boldsymbol{A}) \approx \prod_{t=t_0+1}^{t_0+f} p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_t^0 | \tilde{\boldsymbol{h}}_{t-1}, \boldsymbol{A}), \tag{2}$$

where $\tilde{\boldsymbol{X}}_c^0$ and $\tilde{\boldsymbol{X}}_f^0$ are the noise-free Fourier representations of past and future time series, respectively. $\tilde{\boldsymbol{h}}_{t-1}$ represents past spatio-temporal condition encoded by a spectral recurrent encoder. For each time point $t = t_0 + 1, ..., t_0 + f$, the diffusion process will learn the corresponding backward kernel $p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_t^{k-1} | \tilde{\boldsymbol{x}}_t^k, \tilde{\boldsymbol{h}}_{t-1}, \boldsymbol{A})$ with $k = 1, ..., K$. The objective function of SpecSTG is given by

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{t,k,\tilde{\boldsymbol{x}}_t^0,\boldsymbol{\epsilon}_t} \left\| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_{\boldsymbol{\theta}} \left( \tilde{\boldsymbol{x}}_t^k, k, \tilde{\boldsymbol{h}}_{t-1}, \boldsymbol{A} \right) \right\|^2, \tag{3}$$

where $t$ denotes future time points, $k$ denotes diffusion steps, $\tilde{\boldsymbol{x}}_t^k$ is the disturbed data at time point $t$ and step $k$, and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. It is worth noting that $t$ may also start with $t_0 - c + 1$ instead of $t_0 + 1$ to facilitate the learning of autoregressive dependencies by considering both past and future instances. Lastly, we employ a graph-modified WaveNet architecture (van den Oord et al., 2016) for the denoising network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\cdot)$, which is specially designed for Fourier input. In the rest of this section, we will focus on details of SpecSTG components, training, and inference. The visualization of SpecSTG architecture can be found in Figure 2.

### 4.1 Light-Complexity Spectral Graph Convolution

Diffusion models for STG forecasting usually rely on graph convolution to encode spatial information in the condition of backward kernel (Li et al., 2023; Wen et al., 2023; Liu et al., 2023). Straightforwardly, we adopt the spectral convolution, which typically transforms graph signals to the spectral domain and then processes the data via frequency filtering before they are converted back to the original domain (Defferrard et al., 2016; Kipf & Welling, 2016). However, with SpecSTG, the Fourier representation is already formed as input, so the transform is no longer required in

the convolution. In addition, to ensure that the model pipeline flows in the spectral domain throughout the diffusion learning process, we do not apply reconstruction either.

We choose the Chebyshev convolution (Defferrard et al., 2016), whose filters are formed with Chebyshev polynomials to accelerate computation as $\mathrm{ChebConv}(\boldsymbol{X}) = \boldsymbol{U}\sum_{j=0}^{J-1}\boldsymbol{\phi}_j\mathcal{T}_j(\tilde{\boldsymbol{\Lambda}})\boldsymbol{U}^\intercal\boldsymbol{X}$, for polynomial degrees up to $J-1$, where $\mathcal{T}_j(\tilde{\boldsymbol{\Lambda}}) \in \mathbb{R}^{N\times N}$ is the $j$-th order Chebyshev polynomial evaluated at $\tilde{\boldsymbol{\Lambda}} = 2\boldsymbol{\Lambda}/\lambda_{max} - \boldsymbol{I}_N$ with $\boldsymbol{\phi}_j$ being a learnable coefficient. With Fourier representation $\tilde{\boldsymbol{X}} = \boldsymbol{U}^\intercal\boldsymbol{X}$ as input, we define the modified spectral graph convolution as

$$\mathrm{SpecConv}(\tilde{\boldsymbol{X}}) \coloneqq \sum_{j=0}^{J-1}\boldsymbol{\phi}_j\mathcal{T}_j(\tilde{\boldsymbol{\Lambda}})\tilde{\boldsymbol{X}}. \tag{4}$$

*Remark* 1 (Complexity Analysis). Our spectral convolution has only $\mathcal{O}(N)$ complexity, contrasting to the $\mathcal{O}(N^2)$ complexity of the classic Chebyshev convolution. This significant improvement endows SpecSTG with the advantage of time efficiency over methods using Chebyshev convolution, such as GCRDD. Other methods such as DiffSTG may use the graph convolution in (Kipf & Welling, 2016) as $\mathrm{GCNConv}(\boldsymbol{X}) = \boldsymbol{A}\boldsymbol{X}$, whose complexity is $\mathcal{O}(|\mathcal{E}|)$ where $|\mathcal{E}|$ is the number of edges. As we see in Section 5.1, $|\mathcal{E}|$ is often much larger than $N$ in traffic STGs.

### 4.2 Spectral Recurrent Encoder & Denoising Network

We design SG-GRU as the spectral version of Graph GRU (Seo et al., 2018) to encode past time series and spatial information in graph Fourier domain as follows. For $t = t_0 - c + 1, ..., t_0$:

$$\begin{aligned}
\boldsymbol{z} &= \sigma(\mathrm{SpecConv}(\tilde{\boldsymbol{x}}_t^0)\boldsymbol{W}_{z_1} + \mathrm{SpecConv}(\tilde{\boldsymbol{h}}_{t-1})\boldsymbol{W}_{z_2})\\
\boldsymbol{r} &= \sigma(\mathrm{SpecConv}(\tilde{\boldsymbol{x}}_t^0)\boldsymbol{W}_{r_1} + \mathrm{SpecConv}(\tilde{\boldsymbol{h}}_{t-1})\boldsymbol{W}_{r_2})\\
\boldsymbol{\zeta} &= \tanh(\mathrm{SpecConv}(\tilde{\boldsymbol{x}}_t^0)\boldsymbol{W}_{\zeta_1} + \mathrm{SpecConv}(\boldsymbol{r}\odot\tilde{\boldsymbol{h}}_t)\boldsymbol{W}_{\zeta_2})\\
\tilde{\boldsymbol{h}}_{t+1} &= \boldsymbol{z}\odot\tilde{\boldsymbol{h}}_t + (1-\boldsymbol{z})\odot\boldsymbol{\zeta},
\end{aligned}$$

where $\tilde{\boldsymbol{h}}_0 \in \mathbb{R}^{D_h}$ is a vector of zeors with $D_h$ being the hidden size, $\boldsymbol{W}_{z_1}, \boldsymbol{W}_{r_1}, \boldsymbol{W}_{\zeta_1} \in \mathbb{R}^{1\times D_h}$, $\boldsymbol{W}_{z_2}, \boldsymbol{W}_{r_2}, \boldsymbol{W}_{\zeta_2} \in \mathbb{R}^{D_h\times D_h}$ are learnable weights included in $\boldsymbol{\theta}$, and $\sigma$ is the sigmoid activation function. $\boldsymbol{z}$ and $\boldsymbol{r}$ are known as update gate and reset gate. $\boldsymbol{\zeta}$ is the candidate state storing current information to be updated in the hidden state. In the implementation, we may also input time features $\boldsymbol{\Gamma} = \{\boldsymbol{\gamma}_{t_0-c+1}, ..., \boldsymbol{\gamma}_{t_0+f}\}$ such as day of week and week of month by concatenating them with $\tilde{\boldsymbol{X}}$.

Then, in reminiscent of TimeGrad, we design **S**pectral **G**raph **Wave**Net (SG-Wave) as the $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ of SpecSTG (see details in Appendix C, Figure 5). Besides, we replace some $\mathrm{Conv1d}$ layers with fully connected linear layers for efficient training and sampling. The network takes disturbed data $\tilde{\boldsymbol{x}}_t^k$, diffusion step $k$, hidden condition $\tilde{\boldsymbol{h}}_{t-1}$, and graph adjacency $\boldsymbol{A}$ as inputs, and aims at predicting the noise $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ at step $k$ for time point $t$.

### 4.3 Training and Inference

We compute the Fourier representation of all training data before the training process. Next, we input randomly sampled $\{\tilde{\boldsymbol{X}}_c, \tilde{\boldsymbol{X}}_f\}$ to SG-GRU$_{\boldsymbol{\theta}}$ to obtain hidden states $\tilde{\boldsymbol{h}} = \{\tilde{\boldsymbol{h}}_{t_0}, ..., \tilde{\boldsymbol{h}}_{t_0+f-1}\}$. During training, with a pre-specified noise schedule $\{\beta_1, \beta_2, ..., \beta_K\}$, we randomly sample noise $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and step $k \sim \mathrm{Uniform}(0, K)$ to compute disturbed data $\tilde{\boldsymbol{x}}_t^k$ for $t = t_0+1, t_0+2, ..., t_0+f$ . Finally, we take a gradient step on the objective function in Equation (3). The training algorithm is provided in Appendix D, Algorithm 1. With the denoising network, we can generate samples and make predictions for the forecasting task. The generation process adopts autoregressive sampling, which means we generate samples for each time point one by one. For example, after we generate samples for $t = t_0 + 1$, we feed the sample mean back to the SG-GRU module to compute $\tilde{\boldsymbol{h}}_{t_0+1}$, and then use it to generate samples for $t = t_0 + 2$. Lastly, we convert the predictions back to the original domain via Fourier reconstruction. The sampling and prediction algorithm for a one-time point $\boldsymbol{x}_t^0$ is presented in Appendix D, Algorithm 2.

Table 1: The results of traffic forecasting experiments in a future window of 60 minutes. Average RMSE, MAE, CRPS, and their point values at 15/30/60 minutes are reported. Lower values indicate better forecasting performance. The best results are marked in **bold** and the second best results are underlined. Improvements of SpecSTG on existing methods are shown in percentage.

| Models | RMSE | | | | MAE | | | | CRPS | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Avg. | 15min | 30min | 60min | Avg. | 15min | 30min | 60min | Avg. | 15min | 30min | 60min |
| PEMS04F | | | | | | | | | | | | |
| DeepVAR | 50.59 | 43.90 | 48.76 | 60.46 | 37.74 | 32.97 | 36.72 | 45.87 | 0.2094 | 0.1997 | 0.2108 | 0.2209 |
| TransNVP | 82.74 | 68.26 | 81.70 | 99.81 | 61.85 | 53.06 | 62.25 | 73.49 | 0.2359 | 0.2008 | 0.2377 | 0.2819 |
| TimeGrad | 35.58 | 33.22 | 35.24 | 38.95 | 21.70 | 20.26 | 21.56 | 24.04 | 0.0801 | 0.0747 | 0.0795 | 0.0887 |
| GCRDD | 36.28 | 31.94 | 45.31 | 41.99 | 22.16 | 19.48 | 21.87 | 26.18 | 0.0779 | 0.0689 | 0.0768 | 0.0982 |
| DiffSTG | 37.62 | 34.99 | 36.68 | 43.04 | 24.90 | 22.53 | 24.65 | 29.24 | 0.0904 | 0.0815 | 0.0894 | 0.1077 |
| PriSTI | 33.74 | 33.56 | 33.71 | 37.31 | 22.46 | 21.65 | 22.32 | 25.19 | 0.0772 | 0.0751 | 0.0764 | 0.0870 |
| SpecSTG | **33.15** | **30.07** | **32.81** | **37.29** | **21.53** | **19.29** | **21.39** | **23.29** | **0.0766** | **0.0683** | **0.0761** | **0.0866** |
| Improve. | 1.75% | 5.86% | 2.67% | 0.54% | 0.78% | 0.98% | 0.79% | 3.12% | 0.78% | 0.87% | 0.39% | 0.46% |
| PEMS08F | | | | | | | | | | | | |
| DeepVAR | 41.43 | 35.83 | 39.88 | 49.41 | 27.86 | 23.19 | 27.03 | 34.89 | 0.1291 | 0.1219 | 0.1269 | 0.1412 |
| TransNVP | 67.69 | 60.48 | 68.48 | 76.16 | 51.37 | 49.08 | 52.31 | 58.38 | 0.1802 | 0.1601 | 0.1822 | 0.2073 |
| TimeGrad | 33.09 | 30.17 | 32.53 | 37.51 | 20.47 | 18.24 | 20.06 | 24.24 | 0.0705 | 0.0618 | 0.0695 | 0.0843 |
| GCRDD | 28.83 | 23.91 | 28.10 | 35.68 | 18.72 | 15.52 | 18.35 | 23.99 | 0.0626 | 0.0517 | 0.0617 | 0.0833 |
| DiffSTG | 28.26 | 25.04 | 27.54 | 34.32 | 18.99 | 16.66 | 18.63 | 23.68 | 0.0692 | 0.0609 | 0.0679 | 0.0872 |
| PriSTI | 26.35 | 24.58 | 26.93 | 29.91 | 17.30 | 15.98 | 17.32 | 20.67 | 0.0576 | 0.0539 | 0.0581 | 0.0688 |
| SpecSTG | **25.59** | **22.23** | **24.77** | **29.90** | **17.06** | **14.93** | **16.70** | **20.25** | **0.0572** | **0.0500** | **0.0558** | **0.0680** |
| Improve. | 2.88% | 7.03% | 8.02% | 0.03% | 1.39% | 3.80% | 3.58% | 2.03% | 0.69% | 3.29% | 3.96% | 1.16% |
| PEMS04S | | | | | | | | | | | | |
| DeepVAR | 6.23 | 5.72 | 6.11 | 6.93 | 2.76 | 2.52 | 2.72 | 3.13 | 0.0490 | 0.0450 | 0.0488 | 0.0549 |
| TransNVP | 6.25 | 5.73 | 6.27 | 6.98 | 3.36 | 3.12 | 3.39 | 3.74 | 0.0408 | 0.0382 | 0.0412 | 0.0446 |
| TimeGrad | 5.92 | 5.62 | 5.91 | 6.35 | 2.38 | 2.19 | 2.37 | 2.66 | 0.0307 | 0.0282 | 0.0308 | 0.0345 |
| GCRDD | 4.33 | 3.10 | 4.30 | 5.63 | 1.94 | 1.51 | **1.97** | 2.58 | **0.0245** | **0.0189** | **0.0248** | 0.0329 |
| DiffSTG | 4.46 | 3.24 | 4.46 | 5.72 | 2.15 | 1.66 | 2.20 | 2.83 | 0.0264 | 0.0206 | 0.0267 | 0.0340 |
| PriSTI | 4.42 | 3.31 | 4.67 | 5.60 | 1.96 | 1.54 | 1.99 | 2.62 | 0.0252 | 0.0198 | 0.0258 | 0.0329 |
| SpecSTG | **4.06** | **3.01** | **4.09** | **5.15** | **1.93** | **1.50** | **1.97** | **2.51** | **0.0245** | 0.0192 | 0.0253 | **0.0319** |
| Improve. | 6.24% | 2.90% | 4.88% | 8.04% | 0.52% | 0.66% | 0.00% | 2.71% | 0.00% | - | - | 3.04% |
| PEMS08S | | | | | | | | | | | | |
| DeepVAR | 5.73 | 5.55 | 5.70 | 6.05 | 2.56 | 2.42 | 2.57 | 2.79 | 0.0544 | 0.0534 | 0.0543 | 0.0558 |
| TransNVP | 5.41 | 5.12 | 5.54 | 5.74 | 2.76 | 2.64 | 2.83 | 2.91 | 0.0349 | 0.0334 | 0.0358 | 0.0368 |
| TimeGrad | 4.98 | 4.93 | 4.97 | 5.03 | 1.98 | 1.95 | 1.97 | 2.12 | 0.0267 | 0.0262 | 0.0268 | 0.0272 |
| GCRDD | 3.75 | 2.74 | 3.77 | 4.89 | 1.72 | 1.35 | 1.75 | 2.32 | 0.0223 | 0.0171 | 0.0226 | 0.0301 |
| DiffSTG | 3.97 | 3.20 | 4.07 | 4.82 | 2.36 | 1.91 | 2.43 | 3.04 | 0.0325 | 0.0259 | 0.0335 | 0.0423 |
| PriSTI | 4.22 | 3.02 | 4.39 | 5.20 | 1.70 | 1.29 | 1.80 | 2.15 | **0.0217** | **0.0162** | 0.0230 | 0.0272 |
| SpecSTG | **3.45** | **2.58** | **3.46** | **4.36** | **1.63** | **1.27** | **1.67** | **2.02** | **0.0217** | 0.0170 | **0.0219** | **0.0268** |
| Improve. | 8.00% | 5.84% | 8.22% | 9.54% | 4.12% | 1.55% | 4.57% | 4.72% | 0.00% | - | 0.10% | 1.47% |

# 5 EXPERIMENTS

## 5.1 EXPERIMENT DETAILS

We validate our model on two traffic datasets, PEMS04 and PEMS08 (Guo et al., 2019), collected by the California Transportation Agencies' (CalTrans) Performance Measurement System (PEMS) (Chen et al., 2001). Six probabilistic baselines are considered in our experiments, including four diffusion methods: TimeGrad (Rasul et al., 2021a), GCRDD (Li et al., 2023), DiffSTG (Wen et al., 2023), and PriSTI (Liu et al., 2023); and two non-diffusion methods, DeepVAR (Salinas et al., 2020) and TransNVP (Rasul et al., 2021b). All baselines are state-of-the-art diffusion models proposed in recent years. More experiment details can be found in Appendix E

## 5.2 EXPERIMENT RESULTS

Table 1 presents the results of our traffic forecasting experiments, where the prediction task involves forecasting future time series for a 60-minute horizon based on observations from the past 60 minutes. The table includes numerical values for the average RMSE, MAE, and CRPS over the entire forecast window (all metrics are smaller the better, details can be found in Appendix E). Additionally, it provides corresponding point evaluations assessed at 15, 30, and 60 minutes such that we can evaluate the short and long-term forecasting performance of the models.

**Analysis of deterministic results** SpecSTG consistently achieves top-tier deterministic results across various tasks. In comparison to the second-best model, SpecSTG exhibits an 8.00% improvement in average RMSE for PEMS08S and a 6.24% improvement for PEMS04S. Similarly, the average MAE sees enhancements of 4.12% for PEMS08S and 1.39% for PEMS08F. In addition, our method shows proficiency in both short and long-term deterministic forecasting. Particularly, the

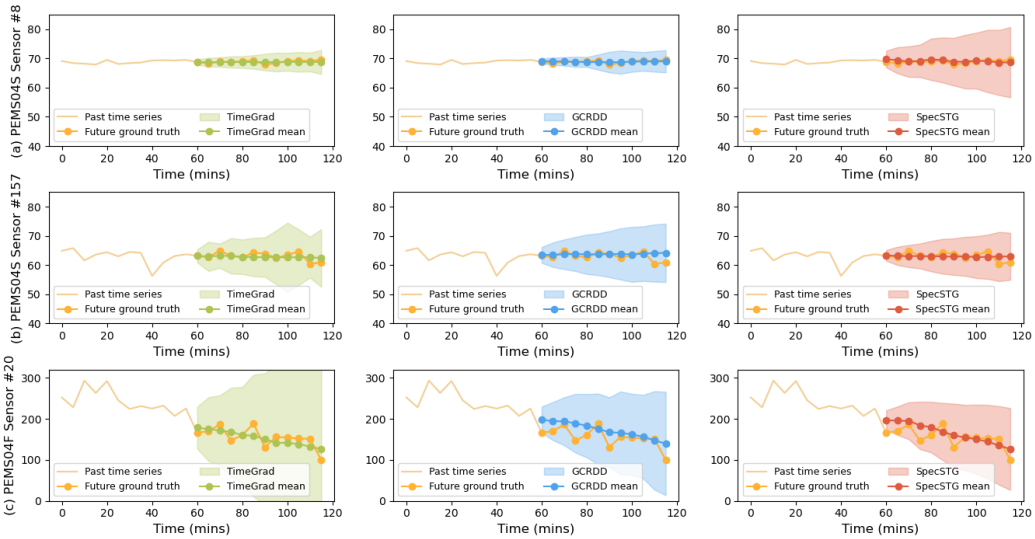


Figure 3: Forecasting visualizations of TimeGrad (green), GCRDD (blue), and SpecSTG (red). (a) and (b) are results on speed data (PEMS04S), while (c) presents results on flow data (PEMS04F).

RMSE improvement of SpecSTG achieves 7.03% on PEMS08F for short-term 15-minute forecasting and 9.54% on PEMS08S for long-term 60-minute forecasting. We also observe improvements in MAE for most results in the table, especially for traffic flow tasks. The superior deterministic performance stems from SpecSTG's unique probabilistic learning process in the spectral domain, leveraging rich global spatial information, which is an acknowledged crucial component in deterministic traffic forecasting (Yu et al., 2018; Fang et al., 2019).

**Analysis of probabilistic results** Regarding the probabilistic metric CRPS, SpecSTG demonstrates an advantage in traffic flow forecasting but not in vehicle speed forecasting. The average CRPS is improved by 0.78% and 0.69% with SpecSTG on PEMS04F and PEMS08F, respectively. However, little improvement is observed on PEMS04S and PEMS08S. Since speed and flow datasets share the same graph structure (e.g., PEMS04S and PEMS04F), we suggest that the inferior probabilistic performance is related to the time series data. A possible explanation is that flow data is often associated with higher variations, thus the systematic variations measured by the Fourier representation are more informative than the speed data. For instance, the standard deviations of data in PEMS08S and PEMS08F are 6.65 and 146.22, respectively. We will further investigate this observation with forecasting visualizations in Subsection 6.1.

## 6 Discussions

In this section, we will visualize SpecSTG's forecasting outcomes compare with two baseline diffusion models. In addition, we will provide analyses on time efficiency and sensitivity to hyperparameters to further show the advantage of SpecSTG. More supplementary discussions can be found in Appendices F and G.

### 6.1 Forecasting Visualizations

Recall that in Subsection 5.2, SpecSTG performs better in probabilistic forecasting on traffic flow data than speed data. Here we further explore this observation by visualizing the forecasting outcomes of TimeGrad, GCRDD, and SpecSTG on PEMS04S and PMES04F (Figure 3). The figure displays the mean and 95% confidence interval (adjusted by time) of estimated future distributions. Our primary objective is to assess whether the forecasting intervals produced by various methods are compatible with the actual future time series. An appropriate interval should capture the future variations while remaining sufficiently narrow to provide meaningful insights.

In traffic speed forecasting, SpecSTG's mean estimation is closer to future time series, but the intervals generated by TimeGrad and GCRDD sometimes better fit the variations in future values. Upon closer examination of the data patterns, we observe that this impact is particularly pronounced in windows with very small variations (Figure 3(a)). In contrast, distributions estimated by SpecSTG at sensors with larger variations Figure 3(b)) exhibit a better ability to capture future uncertainty compared to TimeGrad and GCRDD. This matches our previous hypothesis in Subsection 5.2 that **STG forecasting with larger variations benefits more from the spectral diffusion process**. Because more systematic fluctuations exist in such data, and thus more information can be captured by the Fourier representation, and eventually learned by the diffusion process.

Analogously, in traffic flow forecasting on PEMS04F, a dataset characterized by high systematic variation, SpecSTG demonstrates promising performance by generating both more accurate deterministic predictions and more compatible distributions (Figure 3(c)). This observation reflects the experiment results that SpecSTG achieves outstanding performance with PEMS04F in terms of all metrics.

### 6.2 Time Efficiency

In Figure 4, we compare the training, validation, and sampling time of SpecSTG with three diffusion models for STG forecasting, including GCRDD, DiffSTG, and PriSTI. The training and validation of SpecSTG are clearly faster than other diffusion baselines. Particularly, SpecSTG's training plus validation time is $3.33\times$ of GCRDD, the most efficient method among other existing state-of-the-arts. The validation time of DiffSTG is significantly high because it requires sampling and prediction during validation. To show sampling efficiency, we plot the sampling time per observation (i.e., a future window of 60 minutes) when diffusion steps $K = 50, 100, 200$. We set the batch size of one-shot methods, DiffSTG and PriSTI, as 8 and 16, and report the best results. For autoregressive methods, SpecSTG shows a notable time advantage over GCRDD in sampling. Besides, their sampling time does not vary much with the number of samples $S$. By contrast, the time cost of one-shot methods increases rapidly with the increase of $S$. Although DiffSTG and PriSTI are more efficient when $S$ is small, a small number of samples often cannot present a clear picture of future data distribution.

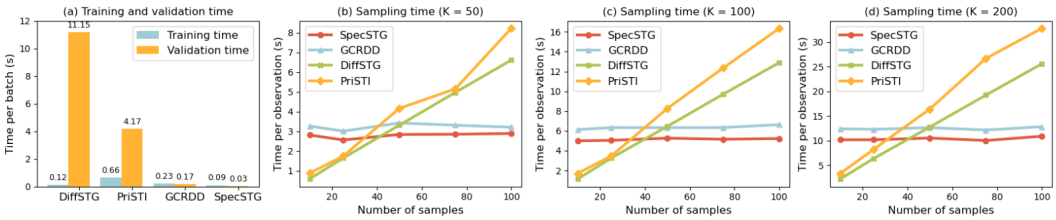


Figure 4: Time efficiency of training, validation, and sampling.

## 7 Concluding Remark

In this paper, we proposed SpecSTG, a spectral diffusion approach for fast probabilistic spatio-temporal traffic forecasting. Our method transforms the entire diffusion learning process to the spectral domain by generating the Fourier representation of future time series instead of the original data. Although we have introduced the autoregressive architecture of SpecSTG, the idea of spectral diffusion can be straightforwardly applied to one-shot methods as well by altering the generative target and graph convolution. Hence, SpecSTG can be regarded as an effective framework for STG forecasting. Experiment results confirm the superior performance of SpecSTG, demonstrating more efficient training and sampling compared to state-of-the-art diffusion methods. Nevertheless, we highlight that SpecSTG may fall short of predicting compatible future distributions when the data have low variations, diminishing the efficacy of spectral measurements of systematic fluctuations.

## REFERENCES


Juan Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *Transactions on Machine Learning Research*, 3, 2023. ISSN 2835-8856.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

Azzedine Boukerche, Yanjie Tao, and Peng Sun. Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems. *Computer Networks*, 182: 107484, 2020.

Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: mining loop detector data. *Transportation Research Record*, 1748 (1):96–102, 2001.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Society, 1997.

Jonathan Crabbé, Nicolas Huynh, Jan Stanczuk, and Mihaela van der Schaar. Time series diffusion in the frequency domain. *International Conference on Machine Learning (ICML)*, 2024.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794, 2021.

Shen Fang, Qi Zhang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Gstnet: Global spatial-temporal network for traffic flow prediction. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2286–2293, 2019.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 922–929, 2019.

William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27953–27965, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 6840–6851, 2020.

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022a.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 8633–8646. PMLR, 2022b.

Junfeng Hu, Xu Liu, Zhencheng Fan, Yuxuan Liang, and Roger Zimmermann. Towards unifying diffusion models for probabilistic spatio-temporal graph learning. *arXiv:2310.17360*, 2023.

Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2023a.

Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, Geoffrey I Webb, Irwin King, and Shirui Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *arXiv:2307.03759*, 2023b.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2016.

Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 21618–21629, 2021.

Ibai Lana, Javier Del Ser, Manuel Velez, and Eleni I Vlahogianni. Road traffic forecasting: Recent advances and new challenges. *IEEE Intelligent Transportation Systems Magazine*, 10(2):93–109, 2018.

Ruikun Li, Xuliang Li, Shiying Gao, ST Boris Choy, and Junbin Gao. Graph convolution recurrent denoising diffusion model for multivariate probabilistic temporal forecasting. In *International Conference on Advanced Data Mining and Applications (ADMA)*, pp. 661–676. Springer, 2023.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 4328–4343, 2022a.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR)*, pp. 1–8, 2018.

Yan Li, Xinjiang Lu, Yaqing Wang, and Dejing Dou. Generative time series forecasting with diffusion, denoise, and disentanglement. In *Advances in Neural Information Processing Systems*, volume 35, pp. 23009–23022, 2022b.

Guojun Liang, Prayag Tiwari, Sławomir Nowaczyk, Stefan Byttner, and Fernando Alonso-Fernandez. Dynamic causal explanation based diffusion-variational graph neural network for spatio-temporal forecasting. *arXiv:2305.09703*, 2023.

Lequan Lin, Zhengkun Li, Ruikun Li, Xuliang Li, and Junbin Gao. Diffusion models for time-series applications: a survey. *Frontiers of Information Technology and Electronic Engineering*, pp. 1–23, 2023.

Mingzhe Liu, Han Huang, Hao Feng, Leilei Sun, Bowen Du, and Yanjie Fu. PriSTI: A conditional diffusion framework for spatiotemporal imputation. In *International Conference on Data Engineering (ICDE)*, pp. 1–10, 2023.

Tianze Luo, Zhanfeng Mo, and Sinno Jialin Pan. Fast graph generation via spectral diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

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

Savinov Nikolay, Chung Junyoung, Binkowski Mikolaj, Elsen Erich, and Oord Aäron van den. Step-unrolled denoising autoencoders for text generation. In *International Conference on Learning Representations*, pp. 1–8, 2022.

Soumyasundar Pal, Liheng Ma, Yingxue Zhang, and Mark Coates. Rnn with particle flow for probabilistic spatio-temporal forecasting. In *International Conference on Machine Learning (ICML)*, pp. 8336–8348. PMLR, 2021.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning (ICML)*, pp. 8857–8868, 2021a.

Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf. Multivariate probabilistic time series forecasting via conditioned normalizing flows. In *International Conference on Learning Representations*, pp. 1–8, 2021b.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241. Springer, 2015.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing (ICONIP)*, pp. 362–373. Springer, 2018.

Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S. Jensen. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *Proceedings of the VLDB Endowment*, 15(11):2733–2746, 2022. ISSN 2150-8097. doi: 10.14778/3551793.3551827.

L. F. Shen and J. Kwok. Non-autoregressive conditional diffusion models for time series prediction. In *International Conference on Machine Learning (ICML)*. PMLR, 2023.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, pp. 2256–2265. PMLR, 2015.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 24804–24816, 2021.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop*, pp. 125, 2016.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with PixelCNN decoders. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

Haomin Wen, Youfang Lin, Yutong Xia, Huaiyu Wan, Qingsong Wen, Roger Zimmermann, and Yuxuan Liang. DiffSTG: Probabilistic spatio-temporal graph forecasting with denoising diffusion models. In *ACM International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL)*, pp. 1–12, 2023.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023.

Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation. *arXiv:2203.09481*, 1, 2022.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

Peiyu Yu, Sirui Xie, Xiaojian Ma, Baoxiong Jia, Bo Pang, Ruigi Gao, Yixin Zhu, Song-Chun Zhu, and Ying Nian Wu. Latent diffusion energy-based model for interpretable text modelling. In *International Conference on Machine Learning*, volume 162, pp. 25702–25720. PMLR, 2022.

Haitao Yuan and Guoliang Li. A survey of traffic prediction: from spatio-temporal data to intelligent transportation. *Data Science and Engineering*, 6:63–85, 2021.

Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1234–1241, 2020.

# A RELATED WORKS

**Generative Diffusion Models** The initial idea of diffusion models was introduced by (Sohl-Dickstein et al., 2015). Then, some improvements proposed by (Ho et al., 2020) endowed them with remarkable practical value, contributing to their conspicuous popularity nowadays. In recent years, diffusion models have demonstrated their power over many existing generative techniques in various real-world applications such as image synthesis (Austin et al., 2021; Dhariwal & Nichol, 2021; Ho et al., 2022a), video generation (Harvey et al., 2022; Ho et al., 2022b; Yang et al., 2022), natural language processing (Li et al., 2022a; Nikolay et al., 2022; Yu et al., 2022), and time series prediction (Rasul et al., 2021a; Li et al., 2022b; Alcaraz & Strodthoff, 2023).

**Diffusion Models for Time Series and STGs** Pioneering diffusion models for time series such as TimeGrad (Rasul et al., 2021a) and TimeDiff (Shen & Kwok, 2023) were originally tailored for multivariate time series forecasting, utilizing sensors as variables but only exploring general dependencies without incorporating graph structural information. Recently, several diffusion models were proposed specifically for STG forecasting. DiffSTG (Wen et al., 2023) incorporates graph structure into the backward kernel with a graph-modified Unet (Ronneberger et al., 2015) architecture. GCRDD (Li et al., 2023), designed in reminiscent of TimeGrad, adopts a graph-enhanced recurrent encoder to produce hidden states from past time series as conditions. Additionally, USTD (Hu et al., 2023) introduces a pre-trained encoder that better captures deterministic patterns via an unsupervised reconstruction task. DVGNN (Liang et al., 2023) is a deterministic model but with a diffusion module to generate dynamic adjacency matrices in its pre-training process. Furthermore, PriSTI (Liu et al., 2023) was initially developed from CSDI (Tashiro et al., 2021) for STG imputation, but with potential for forecasting tasks by masking future data as missing values. We highlight that SpecSTG's novelty lies in its unique spectral diffusion framework that generates graph Fourier representation of future time series, which leverages systematic fluctuations in time series data guided by graph structure to boost forecasting accuracy.

**Spectral diffusion on graphs and time series** The idea of spectral diffusion has been applied in generating graph structure and classic time series. GSDM (Luo et al., 2023) explores the generation of spectral graph structure, i.e., the eigenvalues of graph adjacency matrices, to enhance graph generation quality. Besides, research has shown that generating classic time series in the Fourier domain facilitates diffusion models to better capture the training distribution (Crabbé et al., 2024). Our method, SpecSTG, is the first endeavour to investigate spectral diffusion in generating graph signals and spatio-temporal data.

## B GENERALIZED FOURIER TRANSFORM

The Fourier transform can be naturally generalized to multivariate traffic STGs. Assume that we have a traffic STG with graph signals $\boldsymbol{X}_{\mathcal{G}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_t, ... | \boldsymbol{x}_t \in \mathbb{R}^{N \times D_x}\}$, where $D_x$ is the number of variables. The graph is a univariate STG when $D_x = 1$, and a multivariate STG when $D_x \geq 2$. For a sampled past-future window $\boldsymbol{X} = \{\boldsymbol{x}_{t_0-c+1}, ..., \boldsymbol{x}_{t_0+f}\} \in \mathbb{R}^{N \times D_x \times (c+f)}$, we first split it according to its variables and rewrite it as $\boldsymbol{X} = \{\boldsymbol{x}'_1, ..., \boldsymbol{x}'_d, ..., \boldsymbol{x}'_{D_x} | \boldsymbol{x}'_d \in \mathbb{R}^{N \times (c+f)}\}$. Then, with the Fourier operator $\boldsymbol{U}$, we conduct the transform on each variable matrix $\boldsymbol{x}'_d$ as

$$\tilde{\boldsymbol{x}}'_d = \boldsymbol{U}^{\intercal} \boldsymbol{x}'_d.$$

Hence, the Fourier representation of the sampled window $\boldsymbol{X}$ is $\tilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}'_1, ..., \tilde{\boldsymbol{x}}'_d, ..., \tilde{\boldsymbol{x}}'_{D_x}\} \in \mathbb{R}^{N \times D_x \times (c+f)}$. To convert the Fourier representation of a single variable matrix back to the original domain, we may apply Fourier reconstruction as

$$\boldsymbol{x}'_d = \boldsymbol{U} \tilde{\boldsymbol{x}}'_d,$$

so $\boldsymbol{X} = \{\boldsymbol{U}\tilde{\boldsymbol{x}}'_1, ..., \boldsymbol{U}\tilde{\boldsymbol{x}}'_d, ..., \boldsymbol{U}\tilde{\boldsymbol{x}}'_{D_x}\} \in \mathbb{R}^{N \times D_x \times (c+f)}$.

## C STRUCTURE OF SG-WAVE

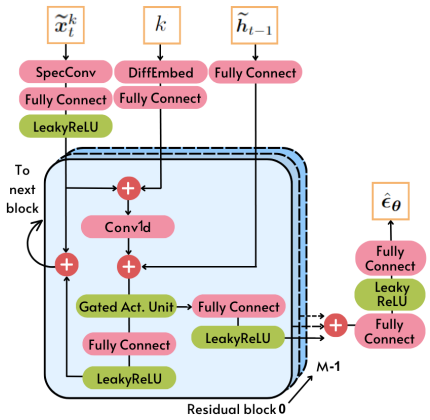


Figure 5: The denoising network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ is a modified WaveNet structure. For computational details please refer to (Rasul et al., 2021a) and (Van den Oord et al., 2016).

# D TRAINING & INFERENCE ALGORITHMS

**Algorithm 1** SpecSTG training

**Input:** The distribution of training data after Fourier transform: $q(\{\tilde{\boldsymbol{X}}_c, \tilde{\boldsymbol{X}}_f\})$; hidden states: $\tilde{\boldsymbol{h}}$; number of diffusion steps: $K$; noise schedule $\{\beta_1, \beta_2, ..., \beta_K\}$, graph: $\mathcal{G}(\mathcal{V}, \mathcal{E}, \boldsymbol{A})$.

**Output:** Optimized denoising network $\boldsymbol{\epsilon_\theta}$.

Sample $\{\tilde{\boldsymbol{X}}_c, \tilde{\boldsymbol{X}}_f\} \sim q(\{\tilde{\boldsymbol{X}}_c, \tilde{\boldsymbol{X}}_f\})$
**while** *Not Convergence* **do**
  **for** $t = t_0 + 1, t_0 + 2, ..., t_0 + f$ **do**
    $k \sim \text{Uniform}(1, K)$, $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
    Compute $\tilde{\boldsymbol{x}}_t^k = \sqrt{\tilde{\alpha}_k}\tilde{\boldsymbol{x}}_t^0 + \sqrt{1 - \tilde{\alpha}_k}\boldsymbol{\epsilon}_t$
  **end for**
  Take gradient step on and do gradient descent for $\theta$

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{t,k,\tilde{\boldsymbol{x}}_t^0,\boldsymbol{\epsilon}_t} \left\| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon_\theta}\left(\tilde{\boldsymbol{x}}_t^k, k, \tilde{\boldsymbol{h}}_{t-1}, \boldsymbol{A}\right) \right\|^2$$

**end while**

**Algorithm 2** SpecSTG sampling and prediction for $\boldsymbol{x}_t^0$

**Input:** Hidden state: $\tilde{\boldsymbol{h}}_{t-1}$; Fourier operator $\boldsymbol{U}$; number of samples: $S$; variance hyperparameter: $\sigma_k$.

**Output:** Prediction $\hat{\boldsymbol{x}}_t^0$.

Randomly generate $S$ samples $\{\tilde{\boldsymbol{x}}_{t,s}^K\}_{s=1}^S \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and do the follows in parallel for all $s$.
**for** $k = K, K-1, ..., 1$ **do**
  $\boldsymbol{e} = \mathbf{0}$ if $k = 1$ else $\boldsymbol{e} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  Compute and update $\tilde{\boldsymbol{x}}_{t,s}^{k-1}$ as

$$\frac{1}{\sqrt{1-\beta_k}}\left(\tilde{\boldsymbol{x}}_{t,s}^k - \frac{\beta_k}{\sqrt{1-\tilde{\alpha}_k}}\boldsymbol{\epsilon_\theta}(\tilde{\boldsymbol{x}}_{t,s}^k, k, \tilde{\boldsymbol{h}}_{t-1}, \boldsymbol{A})\right) + \sigma_k \boldsymbol{e}$$

**end for**
Take average on $S$ samples $\tilde{\boldsymbol{x}}_t^0 = \frac{1}{S}\sum_{s=1}^S \tilde{\boldsymbol{x}}_{t,s}^0$
Compute predictions $\hat{\boldsymbol{x}}_t^0 = \boldsymbol{U}\tilde{\boldsymbol{x}}_t^0$

# E MORE EXPERIMENT DETAILS

## E.1 DATASETS

PEMS04 comprises traffic records from 307 sensors in California's District 04 from Jan 1st to Feb 28th, 2018, while PEMS08 includes data from 170 sensors in District 08 from July 1st to 31st August 2018. Our experiments focus on traffic flow and speed available in both datasets, denoted as "F" and "S" respectively. For instance, "PEMS04F" denotes traffic flow records in the PEMS04 dataset. Each time point covers 5 minutes, with the corresponding value representing the average records during that interval. The speed data are continuous, allowing us to introduce random Gaussian noises. Although traffic flow (i.e., the number of vehicles) is a discrete variable, we still treat it as continuous considering the fact that it contains numerous unique values. More details about the datasets can be found in Table 2.

Table 2: Dataset details.

| Dataset | Type | #Nodes | #Edges | #Time points |
|---|---|---|---|---|
| PEMS04F | Flow | 307 | 680 | 16992 |
| PEMS04S | Speed | | | |
| PEMS08F | Flow | 170 | 548 | 17856 |
| PEMS08S | Speed | | | |

## E.2 IMPLEMENTATION DETAILS

We implement SpecSTG on a single NVIDIA 4090 GPU with 24GB of memory. The model is trained with the Adam optimizer with a learning rate schedule from $5e-4$ to $1e-2$. The maximum number of epochs is 300 for flow data and 50 for speed data with batch size 64. Validation loss is used for model selection. The hyperparameters specific to diffusion models are set as follows. We use the quadratic scheme for noise level $\beta_k$ starting from $\beta_1 = 1e-4$ and tune $\beta_K$ in $[0.1, 0.2, 0.3, 0.4]$. The number of diffusion steps $K$ is selected from $[50, 100, 200]$. The maximum polynomial order in $\mathrm{SpecConv}$ is set as 2. The hidden size $D_h$ is tuned in $[64, 96]$. In SG-Wave, the number of residual blocks $M = 8$ and the residual channel $D_r = 8$. Finally, the number of samples $S$ is set as 100 for all models. For all experiments, we split datasets with 60%/20%/20% train/validation/test proportions and apply Z-score normalization before the Fourier transform. The graph structure is constructed depending on the distance between sensors following (Guo et al., 2019).

## E.3 METRICS

We use three metrics to evaluate the performance of SpecSTG, including two deterministic metrics, Root Mean Squred Error (RMSE) and Mean Absolute Error (MAE), and one probabilistic metric, Continuous Ranked Probability Score (CRPS). RMSE and MAE are adopted to measure the distance between predictions and the ground truth. We use the mean of generated samples as predictions to calculate RMSE and MAE. Given predictions $\hat{\boldsymbol{X}}_f$ at time $t_0$ (after the Fourier reconstruction) and ground truth $\boldsymbol{X}_f$ of one future window, the formulas can be written as

$$\mathrm{RMSE}(\hat{\boldsymbol{X}}_f, \boldsymbol{X}_f) = \sqrt{\frac{1}{f} \sum_{t=t_0+1}^{t_0+f} (\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t)^2}, \tag{5}$$

$$\mathrm{MAE}(\hat{\boldsymbol{X}}_f, \boldsymbol{X}_f) = \frac{1}{f} \sum_{t=t_0+1}^{t_0+f} |\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t|. \tag{6}$$

The final results reported in our experiments are the averages from all available predictive windows in the test set. CRPS is a probabilistic metric that measures the compatibility of the learned probabilistic distribution at each observation (Matheson & Winkler, 1976). Given the cumulative

distribution function (CDF) $F$ of the distribution estimated at observation $x$, CRPS is defined as

$$\mathrm{CRPS}(F^{-1}, x) = \int_0^1 2\left(\kappa - \mathbb{I}_{x<F^{-1}(\kappa)}\right)\left(x - F^{-1}(\kappa)\right) \mathrm{d}\kappa, \tag{7}$$

where $\kappa \in [0, 1]$, $F^{-1}$ is the quantile function, and $\mathbb{I}_{x<F^{-1}(\kappa)}$ is an indicator function which equals to 1 when $x < F^{-1}(\kappa)$ and 0 otherwise. To calculate the integral, we use 100 samples generated at each time point and sensor/node to approximate the corresponding distribution and calculate CRPS following the way defined in (Tashiro et al., 2021). For each future window, we may compute the normalized CRPS at time point $t = t_0 + 1, ..., t_0 + f$ as $\frac{\sum_n \mathrm{CRPS}(F_{t,n}^{-1}, x_{t,n})}{\sum_n |x_{t,n}|}$, where $n = 1, ..., N$ denotes each sensor/node, and $x_{t,n}$ is the value of sensor/node $n$ at time $t$. Likewise, the "CRPS Avg." is computed as $\frac{\sum_{t,n} \mathrm{CRPS}(F_{t,n}^{-1}, x_{t,n})}{\sum_{t,n} |x_{t,n}|}$. We do not adopt $\mathrm{CRPS}_{sum}$ (Rasul et al., 2021a) as a metric because sensors are not regarded as features in our experiments. The results reported for CRPS are also averages over all available predictive windows in the test set.

## F Comparison with Deterministic Baselines

Table 3 presents a comparison between SpecSTG and deterministic methods. Notably, SpecSTG outperforms most classic deterministic approaches, including traditional statistical models like VAR and graph neural networks such as DCRNN and ASTGCN. However, GMAN consistently demonstrates superior performance, particularly in long-term forecasting. Here are two plausible explanations. Firstly, being a deterministic model, GMAN is trained with MAE loss, strengthening the generation of accurate deterministic predictions compared to SpecSTG, which optimizes an implicit probabilistic objective. Secondly, GMAN's sophisticated encoder-decoder architecture and multiple spatio-temporal attention blocks effectively integrate spatial and temporal information, while SpecSTG relies on a single graph-modified GRU encoder for processing spatio-temporal patterns. This observation could guide future work to enhance our model.

Table 3: Comparison between SpecSTG and classic deterministic models on PEMS04F and PEMS08F. Partial results are retrieved from (Shao et al., 2022). Best results are marked in bold, and second best results are underlined.

| Model | RMSE | | | MAE | | |
|---|---|---|---|---|---|---|
| | 15min | 30min | 60min | 15min | 30min | 60min |
| PEMS04F | | | | | | |
| HA | 42.69 | 49.37 | 67.43 | 28.92 | 33.73 | 46.97 |
| VAR | 34.30 | 36.58 | 40.28 | 21.94 | 23.72 | 26.76 |
| FC-LSTM | 33.37 | 39.1 | 50.73 | 21.42 | 25.83 | 36.41 |
| STGCN | 30.76 | 34.43 | 41.11 | 19.35 | 21.85 | 26.97 |
| DCRNN | 31.94 | 36.15 | 44.81 | 20.34 | 23.21 | 29.24 |
| ASTGCN | 31.43 | 34.34 | 40.02 | 20.15 | 22.09 | 26.03 |
| GMAN | **29.32** | **30.77** | **30.21** | **18.28** | **18.75** | **19.95** |
| SpecSTG | <u>30.17</u> | <u>32.81</u> | <u>37.29</u> | <u>19.29</u> | <u>21.39</u> | <u>23.29</u> |
| PEMS08F | | | | | | |
| HA | 34.96 | 40.89 | 56.74 | 23.52 | 27.67 | 39.28 |
| VAR | 29.73 | 30.30 | 38.97 | 19.52 | 22.25 | 26.17 |
| FC-LSTM | 26.27 | 34.53 | 47.03 | 17.38 | 21.22 | 30.69 |
| STGCN | 25.03 | 27.27 | 34.21 | 15.30 | 17.69 | 25.46 |
| DCRNN | 25.48 | 27.63 | 34.21 | 15.64 | 17.88 | 22.51 |
| ASTGCN | 25.09 | 28.17 | 33.68 | 16.48 | 18.66 | 22.83 |
| GMAN | 22.88 | **24.02** | **25.96** | **13.80** | **14.62** | **15.72** |
| SpecSTG | **22.23** | <u>24.77</u> | <u>29.90</u> | <u>14.93</u> | <u>16.70</u> | <u>20.25</u> |

## G Sensitivity Analysis on Hyperparameters

In this sensitivity analysis, we focus on the combination of two very important diffusion hyperparameters: the number of diffusion steps, $K$, and the end of noise schedule, $\beta_K$. Theoretically, the choices of $K$ and $\beta_K$ are relevant to each other. Since the noise level gradually increases from $\beta_1 = 1e-4$ to $\beta_K$ in $K$ diffusion steps, these two hyperparameters control the changing speed and

level of noises in the diffusion process, which is essential for the white noise assumption of diffusion models. The same as the implementation of SpecSTG in our experiments, we set the search spaces of $\beta_K$ and $K$ as $[0.1, 0.2, 0.3, 0.4]$ and $[50, 100, 200]$, respectively. In Figure 6, we use heatmaps to show the change in model performance in terms of RMSE, MAE, and CRPS on PEMS04S with different hyperparameter combinations. We observe that SpecSTG typically performs better with a larger $\beta_K$ (for instance 0.3 or 0.4). The best results appear when $\beta_K = 0.3$ and $K = 50$. It is also worth noting that the forecasting performance of SpecSTG does not vary dramatically with different hyperparameter combinations. This means our method is not very sensitive to hyperparameter selection, alleviating the burden of hyperparameter tuning.

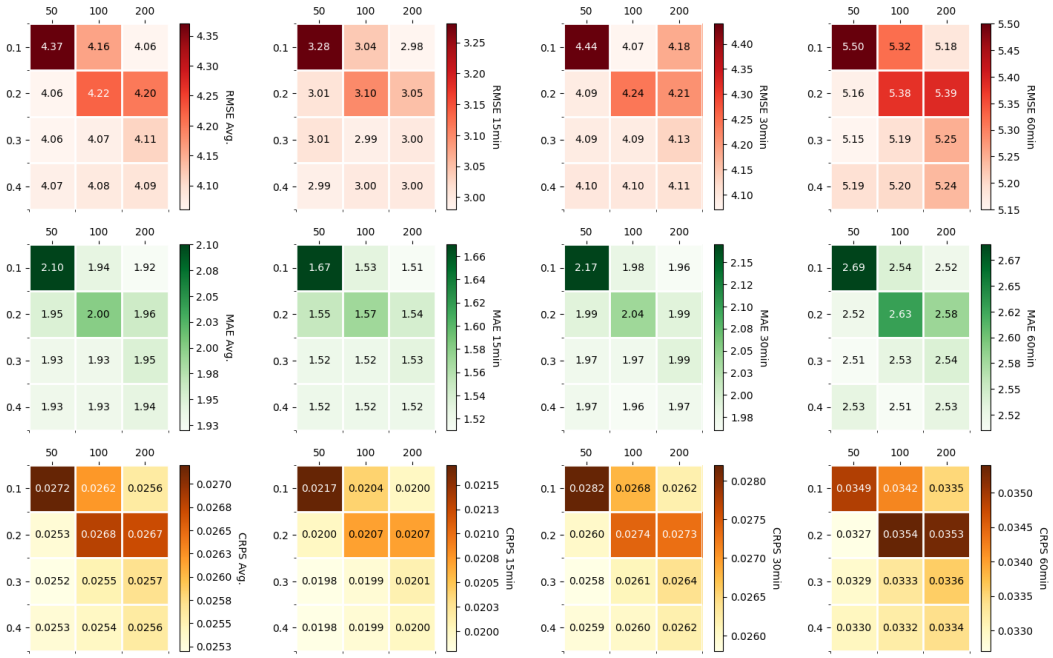


Figure 6: Sensitivity analysis of SpecSTG on key hyperparameters: diffusion steps $K$ and the end of beta schedule $\beta_K$.