Towards Expressive Spectral-Temporal Graph Neural Networks for Time Series Forecasting

Ming Jin, Guangsi Shi, Yuan-Fang Li, Bo Xiong, Tian Zhou, Flora D. Salim, Liang Zhao, *Senior Member, IEEE*, Lingfei Wu, Qingsong Wen, *Senior Member, IEEE*, Shirui Pan, *Senior Member, IEEE*

Abstract—Time series forecasting has remained a focal point due to its vital applications in sectors such as energy management and transportation planning. Spectral-temporal graph neural network is a promising abstraction underlying most time series forecasting models that are based on graph neural networks (GNNs). However, more is needed to know about the underpinnings of this branch of methods. In this paper, we establish a theoretical framework that unravels the expressive power of spectral-temporal GNNs. Our results show that linear spectral-temporal GNNs are universal under mild assumptions, and their expressive power is bounded by our extended first-order Weisfeiler-Leman algorithm on discrete-time dynamic graphs. To make our findings useful in practice on valid instantiations, we discuss related constraints in detail and outline a theoretical blueprint for designing spatial and temporal modules in spectral domains. Building on these insights and to demonstrate how powerful spectral-temporal GNNs are based on our framework, we propose a simple instantiation named *Temporal Graph Gegenbauer Convolution* (TGGC), which significantly outperforms most existing models with only linear components and shows better model efficiency. Our findings pave the way for devising a broader array of provably expressive GNN-based models for time series.

Index Terms—Time series forecasting, graph neural networks, spatio-temporal graphs, graph signal processing, deep learning

1 Introduction

Graph neural networks (GNNs) have achieved considerable success in static graph representation learning for many tasks [1]. Many existing studies, such as STGCN [2] and Graph WaveNet [3], have successfully extended GNNs to time series forecasting. Among these methods, either first-order approximation of ChebyConv [4] or graph diffusion [5] is typically used to model time series relations, where a strong assumption of local homophiles – a property where connected nodes are assumed to have similar features or states – is made [6]. This assumption is inherent in these methods where the graph convolution is approximated by aggregating information from neighboring nodes. Thus, they are only capable of modeling ***positive correlations*** between time series that exhibit strong similarities, and we denote this branch of methods as message-passing-based spatio-temporal GNNs (MP-STGNNs). Nevertheless, how to model real-world multivariate time series with complex spatial dependencies that evolve remains an open question. This complexity is depicted in Fig. 1 within a widely adopted real-world traffic dataset, in which ***differently signed relations*** between time series are evident. For example, in the first window (w_1) of traffic volume, the readings from sensor A and B are positively correlated, but this is not the case in the second window (w_2) of data.

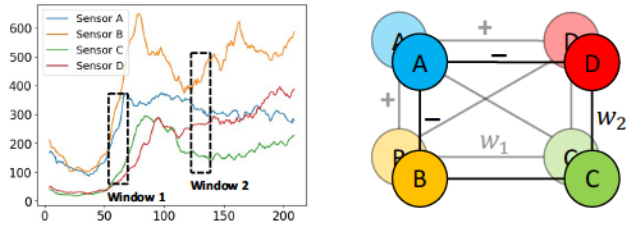


Fig. 1: Differently signed spatial relations between time series in a real-world traffic (PeMS07) dataset. **Left**: Visualization of four randomly selected traffic sensor readings. **Right**: Spatial relations between time series may be different in two windows (e.g., A and B are positively and negatively correlated in windows 1 and 2, respectively), where the (weighted) edges between the sensors normally indicate their correlation strengths but not reflect the signs. Unlike most spatio-temporal GNNs that aggregate the neighborhood information without considering correlation signs, spectral-temporal GNNs go beyond low-pass filtering by learning to aggregate or differentiate such information.

Existing MP-STGNNs often struggle with capturing complex, dynamic relationships in time series data, and

- *Ming Jin and Shirui Pan are with the School of Information and Communication Technology, Griffith University, Gold Coast, Australia. E-mail: mingjinedu@gmail.com, s.pan@griffith.edu.au;*
- *Guangsi Shi and Yuan-Fang Li are with the Department of Data Science and AI, Monash University, Melbourne, Australia. E-mail: {guangsi.shi, yuanfang.li}@monash.edu;*
- *Bo Xiong is with International Max Plank Research School for Intelligent Systems and the University of Stuttgart, Stuttgart, Germany. E-mail: bo.xiong@ki.uni-stuttgart.de;*
- *Tian Zhou is with Alibaba Group, Hangzhou, China. E-mail: tian.zt@alibaba-inc.com;*
- *Flora D. Salim is with the School of Computer Science and Engineering, University of New South Wales, Sydney, Australia. E-mail: flora.salim@unsw.edu.au;*
- *Liang Zhao is with the Department of Computer Science at Emory University, Atlanta, USA. E-mail: liang.zhao@emory.edu.*
- *Lingfei Wu is with Anytime.AI, New York, USA. E-mail: teddy.lfwu@gmail.com.*
- *Qingsong Wen is with Squirrel Ai Learning, Bellevue, US. E-mail: qingsongedu@gmail.com.*
- *Ming Jin and Guangsi Shi contributed equally to this work.*

Corresponding Authors: Qingsong Wen, Shirui Pan.

their limitations stem from a lack of expressiveness, which we define as the ability of a model to accurately represent a wide range of temporal and spatial dependencies within the data. The expressive power of a spatio-temporal GNN is crucial for effective time series forecasting because it determines the model's capacity to capture intricate patterns and relationships that are inherent in real-world time series. Spectral-temporal GNNs (SPTGNNs), as an advanced abstraction of MP-STGNNs, shed light on modeling differently signed time series correlations by approximating graph convolutions with a broad range of graph spectral filters beyond low-pass filtering [7], [8]. This is evidenced by a recent work [9] in multivariate time series forecasting. Although it demonstrated competitive performance, the theoretical foundations of SPTGNNs remain underresearched, which hinders the understanding of SPTGNNs and the development of follow-up research within this model family. Accordingly, in this research, we identify several unresolved fundamental questions:

Q1. What is the general formulation of SPTGNNs?
Q2. How expressive are SPTGNNs in modeling time series data?
Q3. When does this model family fail to generalize well?
Q4. How to design provably expressive SPTGNNs for effective time series forecasting?

In this work, we establish a series of theoretical results summarized in Fig. 2 to answer these questions. We begin by formulating a general framework of SPTGNNs (**Q1**; Sec. 4.1), and then prove its universality of linear models (i.e., linear SPTGNNs are powerful enough to represent arbitrary time series) under mild assumptions through the lens of discrete-time dynamic graphs (DTDGs) [10] and spectral GNNs [13]. We further discuss related constraints from various aspects (**Q3**; Sec. 4.2) to make our theorem useful in practice on ***any*** valid instantiations. After this, we extend the classic color-refinement algorithm [14] on DTDGs and prove that the expressive power of SPTGNNs is theoretically bounded by the proposed temporal 1-WL test (**Q2**; Sec. 4.2). To answer the last question (**Q4**; Sec. 4.3), we prove that under mild assumptions, linear SPTGNNs are sufficient to produce expressive time series representations with orthogonal function bases and individual spectral filters in their graph and temporal frequency-domain models. Our results, for the first time, unravel the capabilities of SPTGNNs and outline a blueprint for designing powerful GNN-based time series models.

Drawing from and to validate these theoretical insights, we present a straightforward, yet effective and novel SPTGNN instantiation, named TGGC (short for *Temporal Graph Gegenbauer Convolution*), that well generalizes the related work in time series forecasting, such as STGCN [2] and StemGNN [9]. Though our primary goal is not to achieve state-of-the-art performance, our method, remarkably, is very efficient and significantly outperforms numerous existing models on several time series benchmarks and forecasting settings ***with minimal designs***. The proposed TGGC building block encompasses only simple linear spatial and temporal frequency-domain models. Comprehensive experiments on synthetic and real-world datasets demonstrate that: **(1)** our approach excels at learning time series relations of different signs compared to MP-STGNNs; **(2)** our design principles, e.g., orthogonal bases and individual filtering, are crucial for SPTGNNs to perform well; **(3)** our instantiation (TGGC) can be readily augmented with nonlinearities and other common model choices. Finally, and more importantly, our findings pave the way for devising a broader array of provably expressive SPTGNNs and thus shed light on subsequent research. Our main contributions in this work are summarized as follows.

Summary of the theoretical results	
SPTGNNs $\succeq$ Linear SPTGNNs	Prop. 1
Linear SPTGNNs is universal under mild conditions	Thm. 1
Linear SPTGNNs $\preceq$ Temporal 1-WL test	Thm. 2
Temporal 1-WL test identifies non-isomorphic nodes	Prop. 2
Conditions of non-isomorphic node pairs in DTDGs	Prop. 3
Constructing graph spectral filters in linear SPTGNNs	Thm. 3
Choosing function bases in temporal space projections	Lem. 1
Linear reduced-order TSFs $\cong$ Linear TSFs	Lem. 2
Constructing temp. spectral filters in linear SPTGNNs	Thm. 4

Fig. 2: An overview of the theoretical results in this work.

- To our knowledge, we are the first to theoretically ground spatio-temporal graph learning in the context of time series forecasting using the frequency analysis. This proves the expressiveness of this family of methods for time series modeling and helps identify situations in which they may fall short.
- We establish a general framework for spectral-temporal graph neural networks (SPTGNNs) and answer the question of how to design provably expressive SPTGNNs for effective time series forecasting. This is exemplified by our simple, yet effective and novel implementation, named *Temporal Graph Gegenbauer Convolution* (TGGC), which contains only linear components and operates entirely in the frequency domain.
- We demonstrate that TGGC can outperform the most representative and related models, and its advanced nonlinear variant (TGGC†) consistently surpasses other competitive counterparts. Our comprehensive evaluation also explicitly demonstrates that our approach excels at learning time series relations of different signs.

The structure of this paper is outlined as follows: Sec. 2 provides an overview of the related work, examining various facets in detail. Sec. 3 delineates the notations and introduces essential background knowledge. Sec. 4 presents a theoretical framework to deepen the understanding of SPTGNNs, addressing the research questions set forth. Sec. 5 showcases our theoretical insights through a novel and straightforward implementation known as TGGC, including its advanced variant TGGC†. Our research findings are evaluated and shared in Sec. 6, and a concise conclusion encapsulating our discoveries can be found in Sec. 7.

2 Related Work

In this section, we provide a concise overview of pertinent literature, encompassing contemporary time series forecast-

ing models, spatio-temporal graph neural networks, and recent advancements in spectral graph neural networks.

2.1 Deep Time Series Forecasting

Time series forecasting has been extensively researched over time. Traditional approaches primarily focus on statistical models, such as vector autoregressive (VAR) [15] and autoregressive integrated moving average (ARIMA) [16]. Deep learning-based approaches, on the other hand, have achieved great success in recent years. For example, recurrent neural network (RNN) and its variants, e.g., FC-LSTM [17], are capable to well model univariate time series. TCN [18] improves these methods by modeling multivariate time series as a unified entity and considering the dependencies between different variables. Follow-up research, such as LSTNet [19] and DeepState [20], proposes more complex models to handle interlaced temporal and spatial clues by marrying sequential models with convolution networks or state space models. Recently, Transformer [21]-based approaches have made great leaps, especially in long-term forecasting [22]. For these methods, an encoder-decoder architecture is normally applied with improved self- and cross-attention, e.g., logsparse attention [23], locality-sensitive hashing [24], and probability sparse attention [25]. As time series can be viewed as a signal of mixed seasonalities, Zhou *et al.* further propose FEDformer [26] and a follow-up work FiLM [27] to rethink how spectral analysis benefits time series forecasting. Nevertheless, these methods do not explicitly model inter-time series relationships (i.e., spatial dependencies).

2.2 Spatio-Temporal Graph Neural Networks

A line of research explores capturing time series relations using GNNs. For instance, DCRNN [28] combines recurrent units with graph diffusion [5] to simultaneously capture temporal and spatial dependencies, while Graph WaveNet [3] interleaves TCN [18] and graph diffusion layers. Subsequent studies, such as STSGCN [29], STFGNN [30], STGODE [31] and G-SWaN [32], adopt similar principles but other ingenious designs to better characterize underlying spatio-temporal clues. However, these methods struggle to model differently signed time series relations since their graph convolutions operate under the umbrella of message passing, which serves as low-pass filtering assuming local homophiles. Some STGNNs, such as ASTGCN [33] and LSGCN [34], directly employ ChebConv [4] for capturing time series dependencies. However, their graph convolutions using the Chebyshev basis, along with the intuitive temporal models they utilize, result in sub-optimal solutions. Consequently, the expressiveness of most STGNNs remains limited. Spectral-temporal graph neural networks (SPTGNNs), on the other hand, first make it possible to fill the gap by (properly) approximating both graph and temporal convolutions with a broad range of filters in spectral domains, allowing for more accurate pattern extraction and modeling. A representative work in this category is StemGNN [9]. However, it faces two fundamental limitations: **(1)** its direct application of ChebConv is sub-optimal, and **(2)** although its temporal FDMs employ orthogonal space projections, they fail to make proper multidimensional and multivariate predictions as discussed.

2.3 Spectral Graph Neural Networks

Spectral graph neural networks are grounded in spectral graph signal filtering, where graph convolutions are approximated by truncated polynomials with finite degrees. These graph spectral filters can be either trainable or not. Examples with predefined spectral filters include APPNP [35] and GraphHeat [36], as illustrated in [7]. Another branch of work employs different polynomials with trainable coefficients (i.e., filter weights) to approximate effective graph convolutions. For instance, ChebConv [4] utilizes Chebyshev polynomials, inspiring the development of many popular spatial GNNs with simplifications [37], [38]. BernNet [7] employs Bernstein polynomials but can only express positive filter functions due to regularization constraints. Another example is [39] that implements the graph convolution with auto-regressive moving average (ARMA) filters. Recently, JacobiConv [13] demonstrated that graph spectral filtering with Jacobi polynomial approximation is highly effective on a wide range of graphs under mild constraints. Although spectral GNNs pave the way for SPTGNNs, they primarily focus on modeling static graph-structured data without the knowledge telling how to effectively convolute on dynamic graphs for modeling time series data.

3 Preliminary

In time series forecasting, given a series of historical observations $\mathbf{X} \in \mathbb{R}^{N\times T\times D}$ encompassing N different D-dimensional variables across T time steps, we aim to learn a function $f(\cdot) : \mathbf{X} \mapsto \hat{\mathbf{Y}}$, where the errors between the forecasting results $\hat{\mathbf{Y}} \in \mathbb{R}^{N\times H\times D}$ and ground-truth $\mathbf{Y}$ are minimized with the following mean squared loss: $\frac{1}{H}\sum_{t=1}^{H} ||\hat{\mathbf{Y}}_t - \mathbf{Y}_t||_F^2$. H denotes the forecasting horizon. In this work, we learn an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ from the input window $\mathbf{X}$ to describe the connection strength between N variables, as in [9]. We use $\mathbf{X}_t := \mathbf{X}_{:,t,:} \in \mathbb{R}^{N\times D}$ to denote the observations at a specific time t, and $\mathbf{X}_n := \mathbf{X}_{n,:,:} \in \mathbb{R}^{T\times D}$ as a time series of a specific variable n with T time steps and D feature dimensions.

Graph Spectral Filtering. For simplicity and modeling the multivariate time series from the graph perspective, we let $\mathcal{G}_t = (\mathbf{A}, \mathbf{X}_t)$ denote an ***undirected*** graph snapshot at a specific time t with the node features $\mathbf{X}_t$. In a graph snapshot, $\mathbf{A}$ and its degree matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ s.t. $\mathbf{D}_{i,i} = \sum_{j=1}^{N} \mathbf{A}_{i,j}$; thus, its normalized graph Laplacian matrix $\hat{\mathbf{L}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{D} - \mathbf{A})\mathbf{D}^{-\frac{1}{2}}$ is symmetric and can be proven to be positive semi-definite. We let its eigendecomposition of $\hat{\mathbf{L}}$ to be $\hat{\mathbf{L}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, where $\mathbf{\Lambda}$ and $\mathbf{U}$ are matrices of eigenvalues and eigenvectors. Below, we define the graph convolution to filter the input signals in the spectral domain w.r.t. the node connectivity.

Definition 1 (Graph Convolution)**.** *Assume there is a filter function of eigenvalues $g(\cdot) : [0, 2] \mapsto \mathbb{R}$, we define the graph convolution on $\mathcal{G}_t$ as filtering the input signal $\mathbf{X}_t$ with the spectral filter:*

$$g(\mathbf{\Lambda}) \star \mathbf{X}_t := \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^\top\mathbf{X}_t. \tag{1}$$

The $\star$ operator is rooted in the graph Fourier transform, where the frequency components of input signal

$\mathbf{U}^\top \mathbf{X}_t$ are transformed with $g(\mathbf{\Lambda})$ before being projected back using $\mathbf{U}$. However, directly computing Eq. 1 can be computationally expensive due to the eigendecomposition involved. To address this, we approximate the ***learnable*** filter function $g_\theta(\mathbf{\Lambda})$ with a truncated K-degree polynomial expansion [40]:

$$g_\theta(\lambda) := \sum_{k=0}^{K} \mathbf{\Theta}_{k,:} P_k(\lambda). \tag{2}$$

Thus, the graph spectral filtering takes the form:

$$\mathbf{U} g_\theta(\mathbf{\Lambda}) \mathbf{U}^\top \mathbf{X}_t := \sum_{k=0}^{K} \mathbf{\Theta}_{k,:} \mathbf{U} P_k(\mathbf{\Lambda}) \mathbf{U}^\top \mathbf{X}_t = \sum_{k=0}^{K} \mathbf{\Theta}_{k,:} P_k(\hat{\mathbf{L}}) \mathbf{X}_t. \tag{3}$$

If we let $g_\theta(\hat{\mathbf{L}}) = \sum_{k=0}^{K} \mathbf{\Theta}_{k,:} P_k(\hat{\mathbf{L}})$, then the graph convolution is redefined as: $g(\mathbf{\Lambda}) \star \mathbf{X}_t := g_\theta(\hat{\mathbf{L}}) \mathbf{X}_t$.

Orthogonal Time Series Representations. Time series can be analyzed in both time and spectral domains. In this work, we focus on modeling time series using sparse orthogonal representations, which are effective in helping disentangle and identify underlying patterns such as periodicities and in reducing modeling complexity. Specifically, for an input signal of the n^{th} variable, we represent $\mathbf{X}_n$ by a set of orthogonal components $\tilde{\mathbf{X}}_n$. As per Lemma 1, numerous orthogonal projections can serve as space projections for our objectives, e.g., [41]. In this study, we employ Fourier transformations by default in our proposed method (Sec. 5).

Definition 2. *Discrete Fourier transformation (DFT) on time series takes measurements at discrete intervals, and transforms observations into frequency-dependent amplitudes:*

$$\tilde{\mathbf{X}}_n(k) = \sum_{t=0}^{T-1} \mathbf{X}_n(t) e^{-2\pi i k t / T}, \ k = \{0, 1, \cdots, T-1\}. \tag{4}$$

Inverse transformation (IDFT) maps a signal from the frequency domain back to the time domain:

$$\mathbf{X}_n(t) = \frac{1}{T} \sum_{k=0}^{T-1} \tilde{\mathbf{X}}_n(k) e^{2\pi i k t / T}, \ t = \{0, 1, \cdots, T-1\}. \tag{5}$$

4 A Theoretical Framework of Spectral-Temporal Graph Neural Networks

We address ***all*** research questions in this section. We first introduce the general formulation of SPTGNNs and then unravel the expressive power of this family of methods within an established theoretical framework. On this basis, we further shed light on the design of powerful SPTGNNs with relevant theoretical justifications and proofs.

4.1 General Formulation

Overall Architecture. We illustrate the general formulation of SPTGNNs in Fig. 3, where we stack M building blocks to capture spatial and temporal dependencies in spectral domains. Without loss of generality, we formulate this framework with the minimum redundancy and a straightforward optimization objective, where common add-ons in prior arts, e.g., spectral attention [26], can be easily incorporated. To achieve time series connectivity without prior knowledge, we directly use the latent correlation layer from [9], as this is not our primary focus.

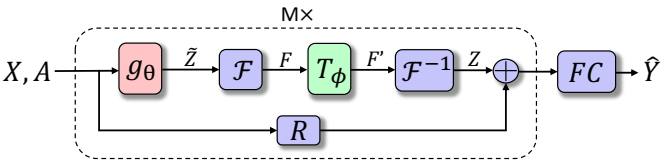


Fig. 3: The general formulation of SPTGNNs with M building blocks to predict future values $\hat{\mathbf{Y}}$ based on historical observations $\mathbf{X}$. We denote $g_\theta(\cdot)$ and $\mathcal{T}_\phi(\cdot)$ as graph and temporal spectral filters. $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ are forward and inverse space projections.

Building Block. From the graph perspective and given the adjacency matrix $\mathbf{A}$, we can view the input signal $\mathbf{X}$ as a particular DTDG with a sequence of regularly-sampled static graph snapshots $\{\mathcal{G}_t\}_{t=0}^{T-1}$, where node features evolve but with fixed graph topology. In a SPTGNN block, we filter $\mathbf{X}$ from the spatial and temporal perspectives in spectral domains with ***graph spectral filters*** (GSFs) and ***temporal spectral filters*** (TSFs). Formally, considering a single dimensional input $\mathbf{X}_d := \mathbf{X}_{:,:,d} \in \mathbb{R}^{N \times T}$, we define a SPTGNN block as follows without the residual connection:

$$\mathbf{Z}_d = \mathcal{T}_{\phi_d}\Big(g_{\theta_d}(\hat{\mathbf{L}}) \mathbf{X}_d\Big) = \mathcal{T}_{\phi_d}\Big(\sum_{k=0}^{K} \mathbf{\Theta}_{k,d} P_k(\hat{\mathbf{L}}) \mathbf{X}_d\Big). \tag{6}$$

This formulation is straightforward. $\mathcal{T}_\phi(\cdot)$ represents TSFs, which work in conjunction with space projectors (detailed in Sec. 4.3) to model temporal dependencies between node embeddings across snapshots. The internal expansion corresponds to GSFs' operation, which embeds node features in each snapshot $\mathcal{G}_t$ by learning variable relations. The above process can be understood in different ways. The polynomial bases and coefficients generate distinct GSFs, allowing the internal K-degree expansion in Eq. 6 to be seen as a combination of different ***dynamic graph profiles*** at varying hops in the graph domain. Each profile filters out a specific frequency signal. Temporal dependencies are then modeled for each profile before aggregation: $\mathbf{Z}_d = \sum_{k=0}^{K} \mathcal{T}_{\phi_d}\Big(\mathbf{\Theta}_{k,d} P_k(\hat{\mathbf{L}}) \mathbf{X}_d\Big)$, which is equivalent to our formulation. Alternatively, dynamic graph profiles can be formed directly in the spectral domain, resulting in the formulation in StemGNN [9] with increased model complexity.

4.2 Expressive Power of Spectral-Temporal GNNs

In this subsection, we develop a theoretical framework bridging spectral and dynamic GNNs, elucidating the expressive power of SPTGNNs for modeling multivariate time series. All proofs are in ***Appendix A***.

Linear GNNs. For a linear GNN on $\mathbf{X} \in \mathbb{R}^{N \times D}$, we define it using a trainable weight matrix $\mathbf{W} \in \mathbb{R}^{D \times d}$ and parameterized spectral filters $g_\theta(\hat{\mathbf{L}})$ as $\tilde{\mathbf{Z}} = g_\theta(\hat{\mathbf{L}}) \mathbf{X} \mathbf{W} \in \mathbb{R}^{N \times d}$. This simple linear GNN can express any polynomial filter functions, denoted as polynomial-filter-most-expressive (PFME), under mild

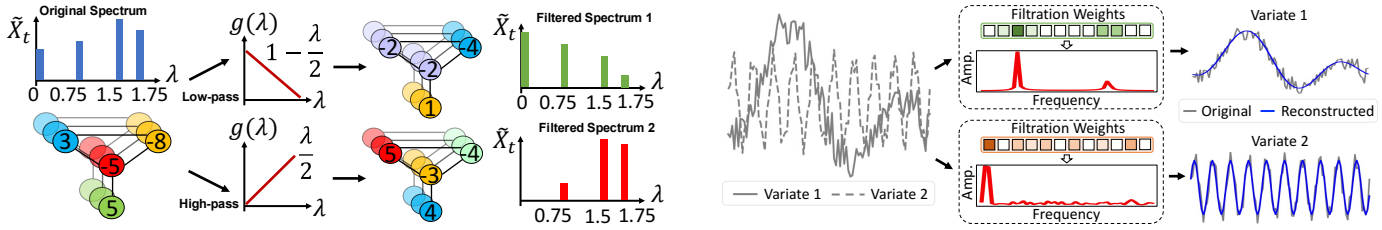


Fig. 4: Multidimensional and multivariate predictions. **Left:** Multidimensional predictions within a snapshot require individual filtration for each output dimension to preserve different information. **Right:** Individual filter is needed to model each single-dimensional time series.

assumptions [13]. Its expressiveness establishes a lower bound for spectral GNNs.

To examine the expressive power of SPTGNNs, we generalize spectral GNNs to model dynamic graphs. For simplicity, we initially consider linear GNNs with a single linear mapping $f_\phi(\cdot)$. To align with our formulation, $\mathcal{T}_\phi(\cdot)$ should also be linear functions; thus, Eq. 6 can be interpreted as a linear GNN extension, where $f_\phi(\cdot)$ depends on historical observations instead of a single graph snapshot. Accordingly, linear SPTGNNs establish a lower bound for the expressive power of SPTGNNs.

Proposition 1. *A* SPTGNN *can differentiate any pair of nodes at an arbitrary valid time that linear* SPTGNNs *can if $\mathcal{T}_\phi(\cdot)$ can express any linear time-variant functions.*

Despite their simplicity, linear SPTGNNs retain the essential operations of spectral filtering, specifically the use of GSFs and TSFs that are integral to SPTGNNs, regardless of the complexity of the model. We begin by defining the ***universal approximation theorem*** for linear SPTGNNs.

Theorem 1. *A linear* SPTGNN *can produce arbitrary dimensional time series representations at any valid time iff:* ***(1)*** $\hat{\mathbf{L}}$ *has no repeated eigenvalues;* ***(2)*** $\mathbf{X}$ *encompasses all frequency components with respect to the graph spectrum;* ***(3)*** $\mathcal{T}_\phi(\cdot)$ *can express any single-dimensional univariate time series.*

Next, we separately explore these conditions and relate them to practical spectral-temporal GNNs.

Multidimensional and Multivariate Prediction. In a graph snapshot, each dimension may exhibit different properties, necessitating distinct filters for processing [13]. An example with two-dimensional predictions is provided in Fig. 4. A simple solution involves using multidimensional polynomial coefficients, as explicitly shown in Eq. 6. A concrete example is presented in [7], where given a series of Bernstein bases, different polynomial coefficients result in various Bernstein approximations, corresponding to distinct GSFs. Similarly, when modeling temporal clues between graph snapshots, either dimension in any variable constitutes a unique time series requiring specific filtering. An example of reconstructing a two-variate time series of one dimension is in Fig. 4. In practice, we use multidimensional masking and weight matrices for each variable, forming a set of different TSFs (Sec. 4.3).

Frequency Components and Eigenvalues. GSFs can only scale existing frequency components of specific eigenvalues. Frequency components sharing the same eigenvalue will be transformed by the same filter. For example, in a graph with repeated eigenvalues due to its topology (e.g., a highly symmetric graph with three- or higher-order automorphisms [13]), multiple frequency components will be scaled by the same $P_k(\lambda)$, thus affecting spectral filtering. Additionally, for each graph snapshot, linear SPTGNNs cannot generate new frequency components if certain frequencies are missing from the original graph spectrum. For instance, in Fig. 4, a frequency component corresponding to $\lambda = 1$ cannot be generated with a spectral filter. Although these two issues are challenging to address, they are rare in real-world attributed graphs [13].

Universal Temporal Spectral Filtering. For a finite-length one-dimensional univariate time series, it can be modeled using a ***frequency-domain model*** (FDM), consisting of sparse orthogonal space projectors and spectral filters. Fig. 4 exemplifies modeling two time series with distinct TSFs. However, this assumes that the time series is well-represented in the transformed space. For instance, it may fail to fully capture non-stationary components or significant non-periodic trends that do not align with the chosen orthogonal projector, such as DFT. Further details are discussed in Sec. 4.3.

Nonlinearity. Nonlinear activation can be applied in both GSFs and TSFs. In the first case, we examine the role of element-wise nonlinearity over the spatial signal, i.e., $\sigma(\mathbf{X}_t)$, enabling frequency components to be mixed w.r.t. the graph spectrum [13]. In the second case, we investigate the role of nonlinearity over the temporal signal by studying its equivalent effect $\sigma'(\cdot)$, as $\sigma(\mathbf{X}_n) = \mathcal{F}^{-1}(\sigma'(\mathcal{F}(\mathbf{X}_n)))$. Here, we have $\sigma'(\tilde{\mathbf{X}}_n) = \mathcal{F}(\sigma(\mathcal{F}^{-1}(\tilde{\mathbf{X}}_n)))$, where different components in $\tilde{\mathbf{X}}_n$ are first mixed (e.g., via Eq. 5) and then element-wise transformed by a nonlinear function $\sigma(\cdot)$ before being redistributed (e.g., via Eq. 4). $\sigma'(\cdot)$ therefore functions as a column-wise nonlinear transformation over all frequency components. Consequently, a similar mixup exists, allowing different components to transform into each other in an orthogonal space, which may help mitigate issues such as missing frequency components from both spatial and temporal perspectives.

Connection to Dynamic Graph Isomorphism. Analyzing the expressive power of GNNs is often laid on graph isomorphism. In this context, we first define the ***temporal***

Weisfeiler-Lehman (WL) test and subsequently establish a connection to linear SPTGNNs in Theorem 2.

Definition 3 (Temporal 1-WL test)**.** *Temporal 1-WL test on discrete-time dynamic graphs $\mathcal{G} := \{\mathcal{G}_t\}_{t=0}^{T-1}$ with a fixed node set $\mathbb{V}$ is defined below with iterative graph coloring procedures:*

① ***Initialization:*** *All node colors are initialized using node features. In a snapshot at time t, we have $\forall v \in \mathbb{V}, c^{(0)}(v,t) = \mathbf{X}_t[v]$. In the absence of node features, all nodes get the same color.*

② ***Iteration:*** *At step l, node colors are updated with an injective (hash) function: $\forall v \in \mathbb{V}, t \in [1,T), c^{(l+1)}(v,t) = \textsc{Hash}(c^{(l)}(v,t), c^{(l)}(v,t-1), \{\!\{c^{(l)}(u,t) : e_{u,v,t} \in \mathbb{E}(\mathcal{G}_t)\}\!\})$. When $t = 0$ or $T = 1$, node colors are refined without $c^{(l)}(v,t-1)$ in the hash function.*

③ ***Termination:*** *The test is performed on two dynamic graphs in parallel, stopping when multisets of colors diverge at the end time, returning non-isomorphic. Otherwise, it is inconclusive.*

The temporal 1-WL test on DTDGs is an extension of the 1-WL test [42]. Based on this, we demonstrate that the expressive power of linear SPTGNNs is bounded by the temporal 1-WL test.

Theorem 2. *For a linear* SPTGNN *with valid temporal FDMs that can express arbitrary 1-dimensional univariate time series and a K-degree polynomial basis in its GSFs, $\forall u, v \in \mathbb{V}, \mathbf{Z}_t[u] = \mathbf{Z}_t[v]$ if $\mathbf{C}^{(K+1)}(u,t) = \mathbf{C}^{(K+1)}(v,t)$. $\mathbf{Z}_t[i]$ and $\mathbf{C}^{(K)}(i,t)$ represent node i's embedding at time t in such a GNN and the K-step temporal 1-WL test, respectively.*

In other words, if a temporal 1-WL test cannot differentiate two nodes at a specific time, then a linear SPTGNN will fail as well. However, this seems to contradict Theorem 1, where a linear SPTGNN assigns any two nodes with different embeddings at a valid time step (under mild assumptions, regardless of whether they are isomorphic or not). For the temporal 1-WL test, it may not be able to differentiate some non-isomorphic temporal nodes and always assigns isomorphic nodes with the same representation/color. We provide examples in Fig. 5. To resolve this discrepancy, we first prove that under mild conditions, i.e., a DTDG has no multiple eigenvalues and missing frequency components, the temporal 1-WL test can differentiate any non-isomorphic nodes at a valid time.

Proposition 2. *If a discrete-time dynamic graph with a fixed graph topology at time t has no repeated eigenvalues in its normalized graph Laplacian and has no missing frequency components in each snapshot, then the temporal 1-WL is able to differentiate all non-isomorphic nodes at time t.*

We further prove that under the same assumptions, all pairs of nodes are non-isomorphic.

Proposition 3. *If a discrete-time dynamic graph with a fixed graph topology has no multiple eigenvalues in its normalized graph Laplacian and has no missing frequency components in each snapshot, then no automorphism exists.*

Therefore, we close the gap and demonstrate that the expressive power of linear SPTGNNs is theoretically bounded by the proposed temporal 1-WL test.

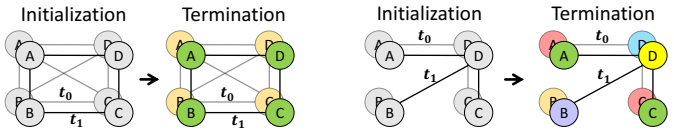


Fig. 5: Two examples of temporal 1-WL test on non-attributive discrete-time dynamic graphs. The left test fails to distinguish non-isomorphic nodes at t_1, e.g., A and C, while the right example demonstrates a successful test.

4.3 Design of Spectral Filters

In this section, we outline a blueprint for designing powerful SPTGNNs, with all proofs available in ***Appendix A***. We initially explore the optimal acquisition of spatial node embeddings within individual snapshots, focusing on the selection of the polynomial basis, $P_k(\cdot)$, for linear SPTGNNs.

Theorem 3. *For a linear* SPTGNN *optimized with mean squared loss, any complete polynomial bases result in the same expressive power, but an orthonormal basis guarantees the maximum convergence rate if its weight function matches the graph signal density.*

This theorem guides the design of GSFs for learning node embeddings in each snapshot w.r.t. the optimization of coefficients. Following this, we discuss the optimal modeling of temporal clues between snapshots in spectral domains. We begin by analyzing the function bases in space projectors.

Lemma 1. *A time series with data points $x_j(t)$ can be expressed by Q uncorrelated components $z_i(t)$ with an orthogonal (properly selected and possibly complex) projector [43], i.e., $x_j(t) = \sum_{i=1}^{Q} e_{ij} z_i(t)$. The eigenvectors e_i are orthogonal and determine the relationship between the data points $x_j(t)$.*

By stabilizing the statistical properties of the input time series, DFT is robust and general enough to represent it in the spectral domain. Additionally, operating on all spectral components is normally unnecessary [26], [44]. Consider a multivariate time series $\mathbf{X}_1(t), \cdots, \mathbf{X}_N(t)$, where each T-length univariate time series $\mathbf{X}_i(t)$ is transformed into a vector $\mathbf{a}_i = (a_{i,1}, \cdots, a_{i,T})^\top \in \mathbb{R}^{T\times 1}$ through a space projection. We form matrix $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_N)^\top \in \mathbb{R}^{N\times T}$ and apply a linear spectral filter as $\mathbf{AW}$. Note that $\mathbf{A}$ does *not* denote the adjacency matrix here. We then randomly select $S < T$ columns in $\mathbf{A}$ using the masking matrix $\mathbf{S} \in \{0,1\}^{S\times T}$, obtaining compact representation $\mathbf{A}' = \mathbf{AS}^\top$ and linear spectral filtration as $\mathbf{A}'\mathbf{W}$. We demonstrate that, under mild conditions, $\mathbf{A}'\mathbf{W}$ preserves most information from $\mathbf{AW}$. By projecting each column vector of $\mathbf{A}$ into the subspace spanned by column vectors in $\mathbf{A}'$, we obtain $P_{\mathbf{A}'}(\mathbf{A}) = \mathbf{A}'(\mathbf{A}')^\dagger\mathbf{A}$. Let $\mathbf{A}_k$ represent $\mathbf{A}$'s approximation by its k largest singular value decomposition. The lemma below shows $||\mathbf{AW} - P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}||_F$ is close to $||\mathbf{W}||_F||\mathbf{A} - \mathbf{A}_k||_F$ if the number of randomly sampled columns S is on the order of k^2.

Lemma 2. *Suppose the projection of $\mathbf{A}$ by $\mathbf{A}'$ is $P_{\mathbf{A}'}(\mathbf{A})$, and the coherence measure of $\mathbf{A}$ is $\mu(\mathbf{A}) = \Omega(k/N)$, then with a high probability, the error between $\mathbf{AW}$ and $P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}$ is bounded*

by $||\mathbf{AW} - P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}||_F \leq (1+\epsilon)||\mathbf{W}||_F||\mathbf{A} - \mathbf{A}_k||_F$ *i*
$S = O(k^2/\epsilon^2)$.

The lemmas above show that in most cases we can ex
press a 1-dimensional time series with **(1)** orthogonal space
projectors and **(2)** a reduced-order linear spectral filter. Fo
practical application on D-dimensional multivariate time
series data, we simply extend dimensions in $\mathbf{S}$ and $\mathbf{W}$.

Theorem 4. *Assuming accurate node embeddings in each snap
shot, a linear* SPTGNN *can, with high probability, produc
expressive time series representations at valid times if its tempora
FDMs consist of:* **(1)** *linear orthogonal space projectors;* **(2**
individual reduced-order linear spectral filters.

4.4 Connection to Related Work

Most of the deep time series models approximate expres
sive temporal filters with deep neural networks and learn
important patterns directly in the time domain, e.g., TCN
[18], where some complex properties (e.g., periodicity) may
not be well modeled. Our work generalizes these methods in
two ways: **(1)** when learning on univariate time series data,
our temporal frequency-domain models guarantee that the
most significant properties are well modeled with high
probability (Lemma 1 and Lemma 2); **(2)** when learning
on multivariate time series data, our framework models
diverse inter-relations between time series (Fig. 1) and
intra-relations within time series with theoretical evidence.
Compared to MP-STGNNs, our framework has two major
advantages: **(1)** it can model differently signed time series
relations by learning a wide range of graph spectral filters
(e.g., low-pass and high-pass), while MP-STGNNs only
capture positive correlations between time series exhibiting
strong similarities; **(2)** on this basis, it can express any mul-
tivariate time series under mild assumptions with provably
expressive temporal frequency-domain models, while MP-
STGNNs approximate effective temporal filtering in time
domains with deep neural networks. Even compared to
STGNNs employing ChebyConv, our proposal generalizes
them well: **(1)** we provide a blueprint for designing effective
graph spectral filters and point out that using Chebyshev
basis is sub-optimal; **(2)** instead of approximating expressive
temporal models with deep neural networks, we detail how
to simply construct them in frequency domains. Therefore,
our results generalize most STGNNs effectively. Although
there are few studies on SPTGNNs, such as StemGNN [9],
we are not only the first to define the general formu-
lation and provide a theoretical framework to generalize
this branch of methods but also free from the limitations
of StemGNN mentioned above. Compared with spectral
GNNs, such as BernNet [7] and JacobiConv [13], our work
extends graph convolution to model dynamic graphs com-
prising a sequence of regularly sampled graph snapshots.

5 Methodology: Temporal Graph Gegenbauer Convolution

In this section, we present a straightforward yet effective
instantiation mainly based on the discussion in Sec. 4.3. We
first outline the basic formulation of the proposed Temporal
Graph Gegenbauer Convolution (TGGC) and then connect

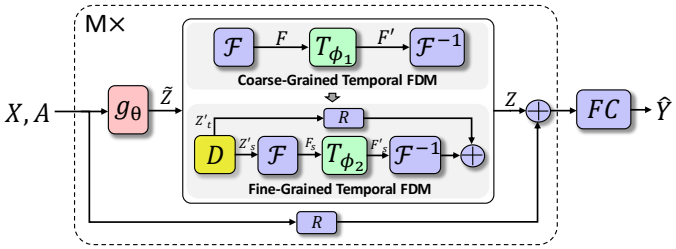


Fig. 6: An illustration of Temporal Graph Gegenbauer Convolution (TGGC), where $g_\theta(\cdot)$, $\mathcal{T}_{\phi_1}(\cdot)$, and $\mathcal{T}_{\phi_2}(\cdot)$ are GSFs and two different TSFs. We use DFT and IDFT as $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ in our implementation. See Fig. 3 for other notations.

it to other common practices. For the sake of clarity, we primarily present the canonical TGGC, which contains ***only*** linear components in its building blocks and strictly inherits the basic formulation presented in Fig. 3. Our primary aim here is not to achieve state-of-the-art performance, but rather to validate our theoretical insights through the examination of TGGC (and its advanced nonlienar variant, TGGC†). Fig. 6 depicts the overall architecture of TGGC, which simply stacks M building blocks as the pattern machine. Each block encompasses a graph convolution and two temporal frequency-domain models.

Graph Gegenbauer Convolution. We implement $P_k(\cdot)$ in GSFs using the Gegenbauer basis due to its **(1)** generality and simplicity among orthogonal polynomials, **(2)** universality regarding its weight function, and **(3)** reduced model tuning expense. See the last paragraph in this section for a detailed justification. The Gegenbauer basis has the form $P_k^\alpha(x) = \frac{1}{k}[2x(k+\alpha-1)P_{k-1}^\alpha(x) - (k+2\alpha-2)P_{k-2}^\alpha(x)]$, with $P_0^\alpha(x) = 1$ and $P_1^\alpha(x) = 2\alpha x$ for $k < 2$. Specifically, $P_k^\alpha(x), k = 0, 1, ...$ are orthogonal on the interval $[-1, 1]$ w.r.t. the weight function $(1-x^2)^{\alpha-1/2}$. Based on this, we rewrite $P_k(\hat{\mathbf{L}})$ in Eq. 6 as $P_k^\alpha(\mathbf{I} - \hat{\mathbf{L}}) = P_k^\alpha(\hat{\mathbf{A}})$, and the corresponding graph frequency-domain model (convolution) is defined as $\sum_{k=0}^K \theta_k P_k^\alpha(\hat{\mathbf{A}})\mathbf{X}$.

Temporal Frequency-Domain Models. When designing temporal FDMs, linear orthogonal projections should be approximately sparse to support dimension reduction, e.g., DFT. For spectral filters, we randomly select S frequency components before filtration. Specifically, for components $\mathbf{f} \in \mathbb{C}^T$ in $\mathbf{F} \in \mathbb{C}^{N\times T\times D}$ along N and D dimensions, we denote sampled components as $\mathbf{f}' := \mathbf{f}_{\mathbb{I}} \in \mathbb{C}^S$, where $\mathbb{I} = \{i_0, \cdots, i_{S-1}\}$ is a set of selection indices s.t. $\forall s \in \{0, \ldots, S-1\}$ and $i_{s-1} < i_s$. This is equivalent to $\mathbf{f}' = \mathbf{f}\hat{\mathbf{S}}^\top$ with $\hat{\mathbf{S}} \in \{0,1\}^{S\times T}$, where $\hat{\mathbf{S}}_{i,s} = 1$ if $i = i_s$. Thus, a standard reduced-order TSF is defined as $\mathbf{f}' = \mathbf{f}\hat{\mathbf{S}}^\top\mathbf{W}$ with a trainable weight matrix $\mathbf{W} \in \mathbb{C}^{S\times S}$.

TGGC Building Block. Suppose $\tilde{\mathbf{Z}} = \sum_{k=0}^K \theta_k P_k^\alpha(\hat{\mathbf{A}})\mathbf{X}$, we discuss two linear toy temporal FDMs. We first consider a basic coarse-grained filtering with the masking and weight matrices $\mathbf{S}_1$ and $\mathbf{\Phi}_1$:

$$\textcircled{1}\ \mathbf{F} = \mathcal{F}(\tilde{\mathbf{Z}}), \quad \textcircled{2}\ \mathbf{F}' = \textsc{Pad}(\mathbf{F}\mathbf{S}_1^\top\mathbf{\Phi}_1), \quad \textcircled{3}\ \mathbf{Z}' = \mathcal{F}^{-1}(\mathbf{F}'). \tag{7}$$

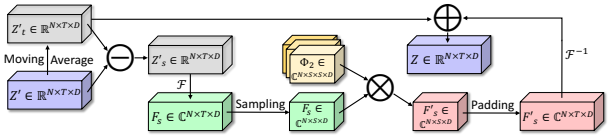


Fig. 7: An illustration of tensor flow in fine-grained temporal frequency-domain models.

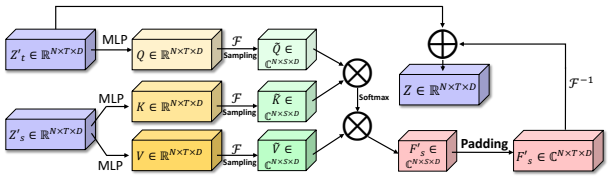


Fig. 8: An illustration of tensor flow in fine-grained temporal frequency-domain models using spectral attention.

Next, we consider an ***optional*** fine-grained filtering based on time series decomposition, where $\mathbf{S}_2$ and $\mathbf{\Phi}_2$ are optimized to further capture the information in detailed signals (e.g., seasonalities) while maintaining global time series profiles (i.e., trends). We formulize this process below and provide an illustration in Fig. 7.

$$\begin{aligned} &① \ \mathbf{Z}'_t, \mathbf{Z}'_s = \textsc{Decomp}(\mathbf{Z}'), \qquad ② \ \mathbf{F}_s = \mathcal{F}(\mathbf{Z}'_s), \\ &③ \ \mathbf{F}'_s = \textsc{Pad}(\mathbf{F}_s \mathbf{S}_2^\top \mathbf{\Phi}_2), \qquad ④ \ \mathbf{Z} = \mathbf{Z}'_t + \mathcal{F}^{-1}(\mathbf{F}'_s). \end{aligned} \tag{8}$$

After stacking multiple blocks, we forecast by transforming time series representations $\mathbf{Z}$. For time series decomposition $\textsc{Decomp}(\cdot)$, we extract the trend information of a time series $\mathbf{x} \in \mathbb{R}^T$ by conducting moving average with a window size w on the input signal, i.e., $\tilde{\mathbf{x}}(t) = \big(\mathbf{x}(t-w+1)+\cdots+\mathbf{x}(t)\big)/w$, where we simply pad the first $w-1$ data points with zeros in $\tilde{\mathbf{x}}$. After this, we obtain detailed time series information (e.g., seasonality) by subtracting the trend from the input signal. For the component padding $\textsc{Pad}(\cdot)$, we pad the filtered signal tensors in both the coarse-grained and fine-grained temporal FDMs with $0+0j$, thereby reshaping them from $\mathbb{C}^{N\times S\times D}$ to $\mathbb{C}^{N\times T\times D}$. We illustrate this in Fig. 7.

Advanced Implementation. Although the proposed TGGC model offers a straightforward and effective demonstration of the concepts, its performance can be enhanced by incorporating nonlinearities and other common modeling choices. TGGC† is different from TGGC in two aspects: **(1)** we incorporate ReLU activation in both graph convolution and temporal FDMs, and **(2)** we implement the spectral filters in fine-grained temporal FDMs with the proposed spectral attention. An illustration is in Fig. 8. We first decompose the input signal via $\mathbf{Z}'_t, \mathbf{Z}'_s = \textsc{Decomp}(\mathbf{Z}')$. After this, we generate the query matrix $\mathbf{Q} = \sigma(\mathbf{\Phi}_{2,1}\mathbf{Z}'_t) \in \mathbb{R}^{N\times T\times D}$, key matrix $\mathbf{K} = \sigma(\mathbf{\Phi}_{2,2}\mathbf{Z}'_s) \in \mathbb{R}^{N\times T\times D}$, and value matrix $\mathbf{V} = \sigma(\mathbf{\Phi}_{2,3}\mathbf{Z}'_s) \in \mathbb{R}^{N\times T\times D}$, where $\sigma(\cdot)$ denotes the ReLU activation, and $\mathbf{\Phi}_{2,i}$ are learnable weights. Then, we project all three matrices into the frequency domain with sparse components, i.e., $\tilde{\mathbf{Q}} = \mathcal{F}(\mathbf{Q})\mathbf{S}_{2,1}^\top \in \mathbb{C}^{N\times S\times D}$, $\tilde{\mathbf{K}} = \mathcal{F}(\mathbf{K})\mathbf{S}_{2,2}^\top \in \mathbb{C}^{N\times S\times D}$, and $\tilde{\mathbf{V}} = \mathcal{F}(\mathbf{V})\mathbf{S}_{2,3}^\top \in \mathbb{C}^{N\times S\times D}$. Finally, we extract informative information with

$$\mathbf{F}'_s = \textsc{Attention}(\tilde{\mathbf{Q}}, \tilde{\mathbf{K}}, \tilde{\mathbf{V}}) = \textsc{Softmax}\Big(\frac{\tilde{\mathbf{Q}}\tilde{\mathbf{K}}^\top}{\sqrt{n_q d_q}}\Big)\tilde{\mathbf{V}}, \tag{9}$$

and we finally have $\mathbf{Z} = \mathbf{Z}'_t + \mathcal{F}^{-1}\big(\textsc{Pad}(\mathbf{F}'_s)\big)$.

Connection to Other Polynomial Bases. We compare our design in TGGC with other common practices in approximating graph convolutions: Monomial, Chebyshev, Bernstein, and Jacobi bases. For non-orthogonal polynomials, our method with the Gegenbauer basis guarantees faster model convergences (Theorem 3) and better empirical performances in most cases. Specifically, for the Monomial basis $(1-\lambda)^k$, it is non-orthogonal for arbitrary choices of weight functions [13]. Although the Bernstein basis $\binom{K}{k}(1-\frac{\lambda}{2})^{K-k}(\frac{\lambda}{2})^k$ is also non-orthogonal, existing studies show that a small conditional number of the Hessian matrix $\kappa(\mathbf{H})$ may still be achieved to enable fast convergence, where $\kappa(\mathbf{H})$ can also be lower than using Monomial basis [45]. While the Bernstein basis is better than the Monomial basis in approximating graph convolutions, our implementation with the Gegenbauer basis guarantees the minimum $\kappa(\mathbf{H})$ to be achieved in most cases; thus, it is more desired. We provide examples in Fig. 9 showing that the weight functions of Gegenbauer polynomials fit graph signal densities well in most cases. We also confirm this in Tab. 6. Compared with other orthogonal polynomials, we know that: **(1)** our basis is a generalization of the second-kind Chebyshev basis; **(2)** though our choice is a particular form of the Jacobi basis, the orthogonality of the Gegenbauer basis is well-posed in most real-world scenarios concerning its weight function. Particularly, the second-kind Chebyshev basis is a particular case of the Gegenbauer basis with $\alpha = 1$ and only orthogonal w.r.t. a particular weight $\sqrt{1-\lambda^2}$. Though the Gegenbauer basis forms a particular case of the Jacobi basis with both of its parameters set to $\alpha - \frac{1}{2}$, we show that the orthogonality of the Gegenbauer basis is well-posed on common real-world graphs w.r.t. its weight function $(1-\lambda^2)^{\alpha-\frac{1}{2}}$ as shown in Fig. 9. Thus, we adopt the Gegenbauer basis as a simpler solution for our purpose with only minor performance degradation (Tab. 6).

6 Evaluation

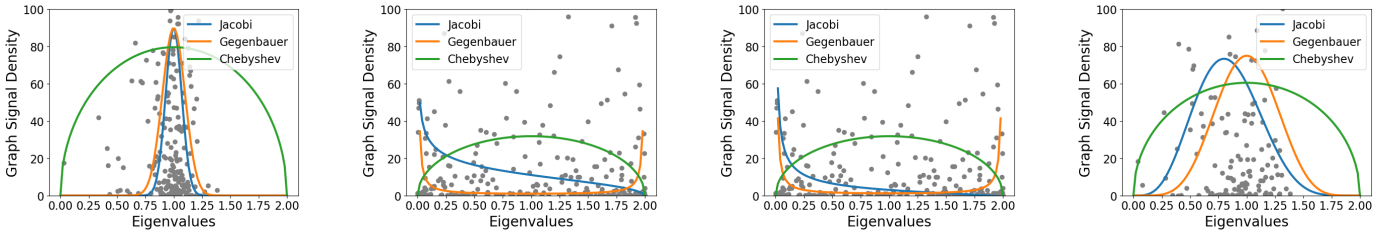
TABLE 1: Statistics of eight different real-world time series datasets used in our work.

Statistic	PeMS03	PeMS04	PeMS07	PeMS08	Electricity	Solar	Weather	ECG
# of time series	358	307	228	170	321	137	21	140
# of data points	26,209	16,992	12,672	17,856	26,304	52,560	52,696	5,000
Sampling rate	5 min	5 min	5 min	5 min	1 hour	10 min	10 min	-
Predefined graph	Yes	Yes	Yes	Yes	No	No	No	No

In this section, we evaluate the effectiveness and efficiency of our proposal on 8 real-world datasets by comparing TGGC and TGGC† with over 20 different baselines. To empirically validate our findings, we further perform extensive ablation studies and present a variety of visualizations using both synthetic and real-world examples.

6.1 Experiment Details

Datasets. For short-term forecasting, we evaluate on 3 time series and 4 traffic datasets commonly used as



(a) PeMS03 Dataset (b) PeMS04 Dataset (c) PeMS07 Dataset (d) PeMS08 Dataset

Fig. 9: Signal density of predefined graphs on different datasets at a randomly selected time step versus the best-fitted weight functions of three different orthogonal polynomials.

TABLE 2: Short-term forecasting results on four traffic benchmarks (*Part I*). We use the **bold** and underline fonts to indicate the best and second-best results. We follow [9] for the experimental setting when compared to TGGC.

Method	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
	PeMS03		*PeMS04*		*PeMS07*		*PeMS08*	
LSTNET [19]	19.07	29.67	24.04	37.38	2.34	4.26	20.26	31.69
DEEPSTATE [20]	15.59	**20.21**	26.50	33.00	3.95	6.49	19.34	27.18
DEEPGLO [46]	17.25	23.25	25.45	35.90	3.01	5.25	15.12	25.22
DCRNN [28]	18.18	30.31	24.70	38.12	2.25	4.04	17.86	27.83
STGCN [2]	17.49	30.12	22.70	35.50	2.25	4.04	18.02	27.83
GWNET [3]	19.85	32.94	26.85	39.70	-	-	19.13	28.16
STEMGNN [9]	14.32	21.64	20.24	32.15	2.14	4.01	15.83	24.93
TGGC **(Ours)**	**13.52**	21.74	**18.77**	**29.92**	**1.92**	**3.35**	**14.55**	**22.73**

TABLE 3: Short-term forecasting results on four traffic benchmarks (*Part II*). We use the **bold** and underline fonts to indicate the best and second-best results. We follow [51] for the experimental setting when compared to TGGC†.

Method	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
	PeMS03		*PeMS04*		*PeMS07*		*PeMS08*	
ASTGCN [33]	17.34	29.56	22.93	35.22	3.14	6.18	18.25	28.06
MSTGCN [33]	19.54	31.93	23.96	37.21	3.54	6.14	19.00	29.15
STG2SEQ [49]	19.03	29.83	25.20	38.48	3.48	6.51	20.17	30.71
LSGCN [34]	17.94	29.85	21.53	33.86	3.05	5.98	17.73	26.76
STSGCN [29]	17.48	29.21	21.19	33.65	3.01	5.93	17.13	26.80
STFGNN [30]	16.77	28.34	20.48	32.51	2.90	5.79	16.94	26.25
STGODE [31]	16.50	27.84	20.84	32.82	2.97	5.66	16.81	**25.97**
TGGC† **(Ours)**	**16.22**	**27.07**	**20.00**	**32.10**	**2.81**	**5.58**	**16.54**	26.10

benchmarks for forecasting models. The PeMS03, 04, 07, and 08 datasets are derived from traffic sensors in highway systems throughout California's metropolitan areas [1]. They are sampled every 5 minutes. The Electricity dataset [2] contains electricity consumption data for 321 customers, sampled hourly. The Weather dataset [3] features the one-year records of 21 meteorological stations installed in Germany, sampled every 10 minutes. The ECG dataset includes 5,000 records from 140 electrocardiograms in the URC time series archive [4]. For long-term forecasting, we utilize the Electricity and Weather datasets, alongside an additional Solar-Energy dataset[2]. This dataset consists of photovoltaic production from 137 sites in Alabama in 2006, with a 10-minute sampling rate.

Baselines. We compare the proposed methods with a list of representative and competitive baselines across different datasets and tasks. In short-term forecasting, our baselines include several deep time series methods: FC-LSTM [47], TCN [18], LSTNet [19], DeepState [20], DeepGLO [46], and SFM [48]. We also compare with a series of spatio-temporal graph neural networks: DCRNN [28], STGCN [2], Graph WaveNet [3], ASTGCN [33], STG2Seq [49], LSGCN [34], STSGCN [29], STFGNN [30], STGODE [31], and StemGNN [9]. In long-term forecasting, we further compare with several state-of-the-art Transformer-based models: FiLM [27], FEDformer [26], Autoformer [50], Informer [25], LogTrans [23], and Reformer [24].

Implementation Details. All experiments are conducted on 4 × Nvidia RTX 2080 Ti 11GB GPUs. We utilized the Mean Absolute Forecasting Errors (MAE) and Root Mean Squared Forecasting Errors (RMSE) as our primary metrics. Our results are averaged over 5 runs.

For short-term forecasting, our default data splits are 60%-20%-20% for the PeMS03, 04, and 08 datasets, and 70%-20%-10% for the others as per [9]. Min-max normalization is applied to the ECG dataset, while Z-Score normalization is employed for the rest. Using the past 1-hour observations from four traffic datasets, we predict the next 15-minute traffic volumes or speeds, a setting consistently adopted in ablation studies and model efficiency comparisons. For the Electricity dataset, the prior 24-hour readings forecast the subsequent 1-hour consumption. The Solar-Energy dataset employs the previous 4 hours of data to predict the next half-hour production. The ECG dataset has a window size of 12 and a forecasting horizon of 3. All baseline configurations are the same as in [9]. In Tab. 3, we compare to a series of competitive STGNNs and follow the setup in [51] with minor differences: we use the 60%-20%-20% data split across all four datasets, and leverage the past 1-hour observations to predict the next 1-hour traffic volume or speed.

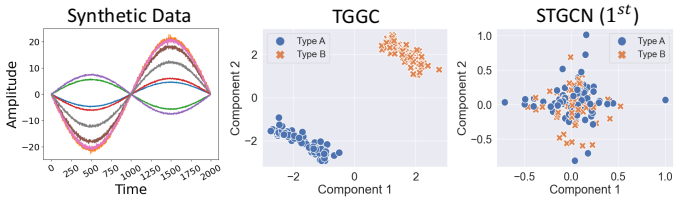
TABLE 4: Efficiency comparison of representative models: Trainable parameters (M), time-per-epoch (s), and total training time (min); ♢ indicates significantly larger values compared to other methods.

Method	*PeMS03*	*PeMS04*	*PeMS07*	*PeMS08*
LSTNET	0.4/2.2/3.8	0.3/1.3/2.3	0.2/0.9/1.5	0.2/1.0/1.7
DEEPGLO	0.6/14/8.3	0.6/8.5/6.0	0.3/6.2/6.6	0.3/5.9/4.2
DCRNN	OOM	0.4/♢/♢	0.4/♢/♢	0.4/♢/♢
STGCN	0.3/25/13	0.3/14/6.9	0.2/8.6/4.1	0.2/8.0/5.0
STEMGNN	1.4/17/24	1.3/9.0/13	1.2/6.0/8.0	1.1/6.0/9.0
TGGC **(Ours)**	0.4/12/21	0.3/6.2/11	0.2/3.9/7.2	0.1/4.2/8.4

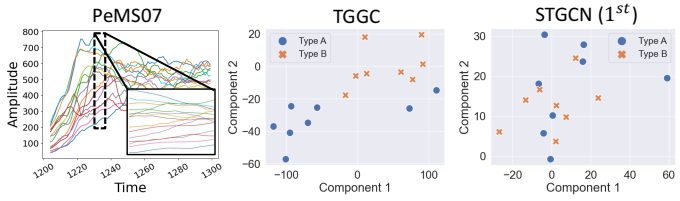
1. https://pems.dot.ca.gov/
2. https://github.com/laiguokun/multivariate-time-series-data
3. https://github.com/zhouhaoyi/Informer2020
4. https://www.cs.ucr.edu/~eamonn/time_series_data/

TABLE 5: Long-term forecasting results on three time series benchmarks. We follow [27] for the experimental setting when compared to TGGC†.

Method		TGGC† (Ours)		FiLM [27]		FEDFormer [26]		Autoformer [50]		Informer [25]		LogTrans [23]		Reformer [24]	
Metric		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Electricity	96	0.293	0.425	**0.267**	**0.392**	0.297	0.427	0.317	0.448	0.368	0.523	0.357	0.507	0.402	0.558
	192	0.303	0.440	**0.258**	**0.404**	0.308	0.442	0.334	0.471	0.386	0.544	0.368	0.515	0.433	0.590
	336	0.313	0.470	**0.283**	**0.433**	0.313	0.460	0.338	0.480	0.394	0.548	0.380	0.529	0.433	0.591
Weather	96	**0.235**	**0.408**	0.262	0.446	0.296	0.465	0.336	0.515	0.384	0.547	0.490	0.677	0.596	0.830
	192	**0.286**	**0.468**	0.288	0.478	0.336	0.525	0.367	0.554	0.544	0.773	0.589	0.811	0.638	0.867
	336	**0.317**	**0.515**	0.323	0.516	0.380	0.582	0.395	0.599	0.523	0.760	0.652	0.892	0.596	0.799
Solar	96	**0.242**	**0.443**	0.311	0.557	0.363	0.448	0.552	0.787	0.264	0.469	0.262	0.467	0.255	0.451
	192	**0.263**	**0.470**	0.356	0.595	0.354	0.483	0.674	0.856	0.280	0.487	0.284	0.489	0.274	0.475
	336	**0.271**	**0.478**	0.370	0.628	0.372	0.518	0.937	1.131	0.285	0.496	0.295	0.512	0.278	0.491



(a) Visualization of learned embeddings w.r.t. different time series correlations on a synthetic dataset. Types A and B represent series groups with opposing trends (e.g., pink vs. green).



(b) Visualization of learned embeddings w.r.t. different time series correlations on PeMS07 dataset. Types A and B represent series groups with opposing trends.

Fig. 10: Evaluation of learning differently signed time series relations.

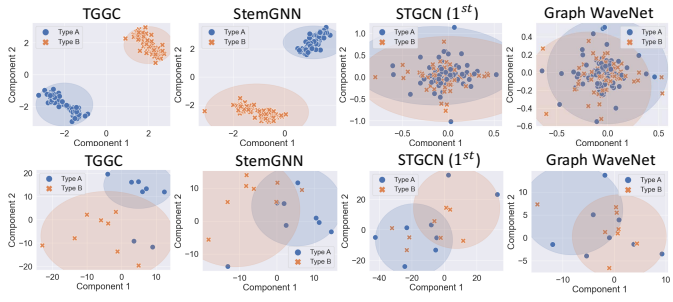


Fig. 11: Additional comparison between learned embeddings w.r.t. different time series correlations on a synthetic dataset presented in Fig. 10a **(upper)** and a subset of traffic recordings in PeMS07 dataset as in Fig. 10b **(lower)**. Types A and B denote series groups exhibiting opposing trends. Differently colored shading indicates distinct clusters.

For long-term forecasting, we adhere to [27], using a 70%-10%-20% split for all datasets. The Electricity dataset uses 4 days of past observations to predict the next 4, 8, and 14 days. Both the Solar-Energy and Weather datasets employ 16-hour past data to predict the subsequent 16, 32, and 56 hours. Hyperparameter settings are detailed in ***Appendix B***.

6.2 Main Results.

We evaluate against related work in terms of model effectiveness (Tab. 2, Tab. 3, and Tab. 5) and efficiency (Tab. 4), showcasing the potential of SPTGNNs for time series forecasting. We compare our vanilla instantiation (TGGC) with the most pertinent and representative works in Tab. 2 on four traffic benchmarks. Next, we assess our method (TGGC†, the nonlinear version of TGGC) against a series of competitive STGNNs in Tab. 3. The main differences here are the data splits and forecasting horizons we followed in [9] and [51] as mentioned above. Additionally, we report long-term forecasting results in Tab. 5, comparing our approach with state-of-the-art Transformer models.

Our method consistently outperforms most baselines by significant margins. In short-term forecasting, TGGC and TGGC† achieve average performance gains of **7.3%** and **1.7%** w.r.t. the second-best results. Notably, we observe a significant improvement (∼**8%**) over StemGNN [9], a special case of our method with nonlinearities. Considerable enhancements are also evident when compared to AST-GCN [33] (∼**9%**) and LSGCN [34] (∼**7%**), which primarily differ from StemGNN in temporal dependency modeling and design nuances. In long-term forecasting, our method further exhibits impressive performance, outperforming the second-best results by about **3.3%**. These results indicate that even simple, yet appropriately configured SPTGNNs (discussed in Sec. 4.3) are potent time series predictors. In Tab. 4, we examine our method's efficiency by comparing TGGC to representative baselines. We find that TGGC forms the simplest and most efficient SPTGNN to date compared with StemGNN. In comparison to other STGNNs, such as STGCN [2] and DCRNN [28], our method also exhibits superior model efficiency across various aspects. Though time series models like LSTNet [19] are faster in training, they do not have spatial modules and thus less effective.

6.3 Evaluation of Modeling Time Series Dependencies.

Our method, in contrast to most GNN-based approaches, excels at learning different signed spatial relations between

TABLE 6: Ablation study results. We use the **bold** and underline fonts to denote the best and second-best results in each ablation block, respectively.

Variant	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
	PeMS03		*PeMS04*		*PeMS07*		*PeMS08*	
A.1 Monomial	27.64	43.52	59.41	120.18	5.68	8.73	29.36	43.37
A.2 Bernstein	27.38	43.17	55.17	105.13	5.57	8.64	27.57	40.28
A.3 Chebyshev	13.56	21.84	18.78	29.89	1.94	3.37	14.36	22.93
A.4 Gegenbauer	13.52	21.74	18.77	29.92	1.92	3.35	14.35	22.73
A.5 Jacobi	**13.14**	**21.51**	**18.64**	**29.44**	**1.91**	**3.34**	**14.29**	**22.15**
TGGC **(Ours)**	**13.52**	21.74	**18.77**	**29.92**	**1.92**	**3.35**	**14.55**	**22.73**
B.1 w/o MD-F	14.07	21.82	19.07	30.34	2.05	3.45	15.14	23.49
B.2 w/o MD-F†	13.62	21.73	18.92	30.11	1.96	3.40	15.06	23.25
B.3 w/o MV-F	13.80	21.77	19.12	30.50	1.97	3.44	15.12	23.58
B.4 w/o O-SP	13.72	21.75	19.10	30.42	2.03	3.53	15.38	28.84
B.5 w/o C-FDM	13.83	**21.47**	19.15	30.52	2.01	3.47	15.18	23.71
B.6 w/o F-FDM	13.71	21.59	18.81	29.96	1.93	3.39	14.88	23.10
TGGC† **(Ours)**	**13.39**	**21.34**	**18.41**	**29.39**	**1.84**	**3.28**	**14.38**	**22.43**
C.1 w/o NL	13.75	21.43	18.63	29.70	1.92	3.36	14.94	24.93
C.2 w/o S-Attn	13.42	21.38	18.50	29.53	1.86	3.30	14.43	22.54

time series (Fig. 1). Predetermined or learned graph topologies typically reflect the strength of underlying connectivity, yet strongly correlated time series might exhibit distinct properties (e.g., trends), which MP-STGNNs often struggles to model effectively. To substantiate our claims, we first visualize and compare the learned TGGC and STGCN(1^{st}) [2] representations (e.g., $\mathbf{Z}$ mentioned after Eq. 8 in our method) using t-SNE [52] on two synthetic time series groups with positive and negative correlations. We create two groups of time series, each with a length of 2000. For the first group, we generate a sinusoidal signal and develop 100 distinct instances with varying amplitudes and injected random noise. Similarly, we generate another group of data based on cosinusoidal oscillation. Fig. 10a reveals that STGCN(1^{st}) fails to differentiate between the two correlation groups. This is because methods like STGCN(1^{st}) aggregate neighborhood information with a single perspective (i.e., low-pass filtration). We further examine the learned embeddings of two groups of randomly sampled time series with opposing trends between TGGC and STGCN(1^{st}) on a real-world traffic dataset (Fig. 10b), where similar phenomena can be observed.

Additional evaluation can be found in Fig. 11, where we compare the learned time series embeddings from two SPTGNNs (i.e., Ours and StemGNN [9]) and MP-STGNNs (i.e., STGCN(1^{st}) [2] and Graph WaveNet [3]). It is evident that SPTGNNs excel in learning different signed spatial relations between time series, yielding significantly distinct representations with much higher clustering purities in both cases, further substantiating our claims in the main text.

6.4 Ablation Studies.

We perform ablation studies from three perspectives. First, we evaluate the GSFs in TGGC with different polynomial bases (A.1 to A.5) in the first block. Next, we examine other core designs in the second block: B.1 and B.2 apply identical polynomial coefficients in GSFs and trainable weights in temporal FDMs along D dimensions, respectively. B.3 utilizes the same set of TSFs across N variables. B.4 replaces orthogonal space projections with random transformations. B.5 and B.6 separately remove the coarse-grained and fine-grained temporal FDMs. Lastly, we evaluate add-ons that make TGGC†. C.1 eliminates nonlinearities, and C.2 disables the spectral attention.

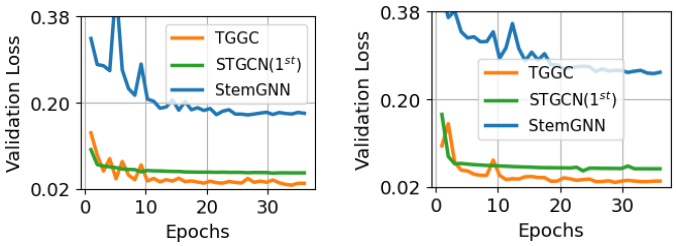
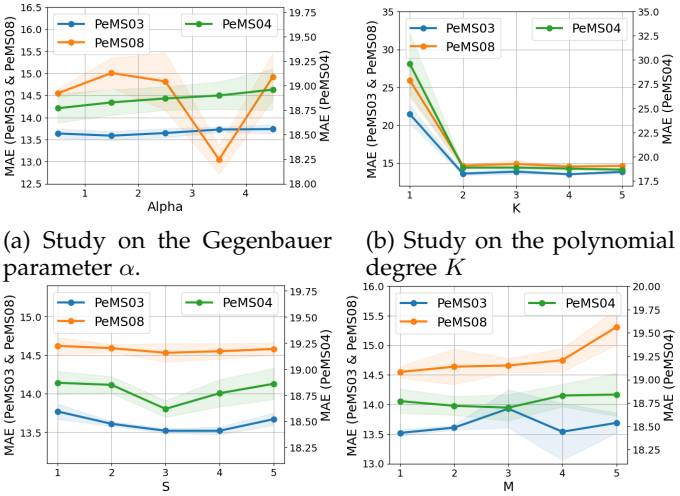


Fig. 12: Model convergence comparison on PeMS07 dataset. **Left**: $lr = 0.01$. **Right**: $lr = 0.001$.

In the first block of results, we validate the discussion in Sec. 5: **(1)** orthogonal polynomials (A.3 to A.5) yield significantly better performance than non-orthogonal alternatives (A.1 and A.2); **(2)** although the performance gaps between A.3 to A.5 are minor, polynomial bases with orthogonality that hold on more general weight functions tend to result in better performances. The results of B.1 to B.3 support the analysis of multidimensional and multivariate predictions in Sec. 4.2, with various degradations observed. In B.4, we see an average **4%** MAE and **7%** RMSE reduction, confirming the related analysis in Sec. 4.3. The results of B.5 and B.6 indicate that both implementations (Eq. 7 and Eq. 8) are effective, with fine-grained temporal FDMs icing on the cake. In the last block, we note a maximum **3.1%** and **0.8%** improvement over TGGC by introducing nonlinearities (C.2) and spectral attention (C.1). Simply combining both leads to even better performance (i.e., TGGC†).

6.5 Additional Analysis

Model Convergence. We compare model convergence between STGCN(1^{st}) [2], StemGNN [9], and TGGC across two scenarios with different learning rates in Fig. 12. Our implementation with Gegenbauer bases has the



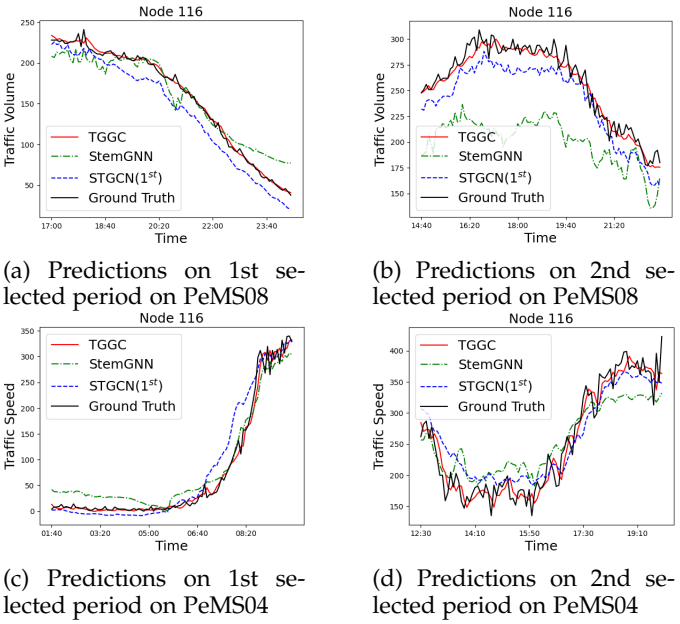
(a) Study on the Gegenbauer parameter α.

(b) Study on the polynomial degree K

(c) Study on the number of selected modes S.

(d) Study on the number of building blocks M.

Fig. 13: Study on important parameters in TGGC.



(a) Predictions on 1st selected period on PeMS08

(b) Predictions on 2nd selected period on PeMS08

(c) Predictions on 1st selected period on PeMS04

(d) Predictions on 2nd selected period on PeMS04

Fig. 14: Showcases of model forecasts on different datasets.

fastest convergence rate in both cases, further confirming Theorem 3 with ablation studies. Also, as anticipated, STGCN(1^{st}) is more tractable than StemGNN w.r.t. the model training due to certain relaxations but at the cost of model effectiveness.

Parameter Sensitivity. The results of parameter study are in Fig. 13, and we have the following observations: **(1)** adjusting the α in Gegenbauer polynomials impacts model performance variably across datasets. For instance, a smaller α is generally favored in three PeMS datasets, as depicted in Fig. 13a; **(2)** the polynomial degree should not be too small to avoid information loss, as demonstrated by the poor performance when $K = 1$ in Fig. 13b. In practice, we set K between 3 and 5, as higher degrees do not bring additional performance gains; **(3)** similarly, setting S too small or too large is not desirable, as shown in Fig. 13c. This is to prevent information loss and mitigate the impact of noise; **(4)** stacking more building blocks seems unnecessary; even a single TGGC block already yields competitive performance, according to Fig. 13d.

Showcases. Forecasting visualizations are presented in Fig. 14. We juxtapose TGGC with StemGNN and STGCN(1^{st}) using the PeMS08 and PeMS04 datasets. The visualizations capture forecasts from two randomly selected sensors during distinct periods from the test sets. In every instance, TGGC markedly surpasses the other two baselines.

Additional Experiments. Refer to ***Appendix C*** for details. Specifically, **(1)** we evaluate TGGC against additional baselines on other time series benchmarks; **(2)** we provide the statistics of our forecasting results.

7 Conclusion

In this study, we provide the general formulation of spectral-temporal graph neural networks (SPTGNNs), laying down a theoretical framework for this category of methods. The key insights derived include: **(1)** under modest assumptions, SPTGNNs can achieve universality; **(2)** the use of orthogonal bases and individual spectral filters are pivotal in crafting potent GNN-based time series models. To validate our theoretical findings, we introduce an innovative yet straightforward spectral-temporal graph neural network (TGGC) and its enhanced variant (TGGC†). Both consistently surpass the majority of baseline methods across diverse real-world benchmarks. Our findings shed light on the follow-up research and pave the way for devising a broader array of provably expressive SPTGNNs. A limitation of our approach is that SPTGNNs may face challenges in handling specific types of time series data, such as the underlying graph topology is highly symmetric and node attributes exhibit strong non-stationary or non-periodic behavior. These cases, while rare, can pose difficulties for our method and will necessitate further refinement of the development of more adaptive models. In addition to addressing this limitation, future research will also explore specific scenarios such as time-evolving graph structures and investigating the applicability of our theories to other tasks, such as time series classification and anomaly detection.

Acknowledgments

This research was partly funded by Australian Research Council (ARC) under grants FT210100097 and DP240101547 and the CSIRO – National Science Foundation (US) AI Research Collaboration Program.

References

[1] W. L. Hamilton, "Graph representation learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 14, no. 3, pp. 1–159, 2020.

[2] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *IJCAI*, 2018, pp. 3634–3640.

[3] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *IJCAI*, 2019, pp. 1907–1913.

[4] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NeurIPS*, 2016, pp. 3837–3845.

[5] J. Klicpera, S. Weißenberger, and S. Günnemann, "Diffusion improves graph learning," in *NeurIPS*, 2019, pp. 13 366–13 378.

[6] S. Li, D. Kim, and Q. Wang, "Beyond low-pass filters: Adaptive feature propagation on graphs," in *ECML PKDD*. Springer, 2021, pp. 450–465.

[7] M. He, Z. Wei, H. Xu *et al.*, "Bernnet: Learning arbitrary graph spectral filters via bernstein approximation," in *NeurIPS*, vol. 34, 2021, pp. 14 239–14 251.

[8] T. Derr, Y. Ma, and J. Tang, "Signed graph convolutional networks," in *ICDM*. IEEE, 2018, pp. 929–934.

[9] D. Cao, Y. Wang, J. Duan, C. Zhang, X. Zhu, C. Huang, Y. Tong, B. Xu, J. Bai, J. Tong *et al.*, "Spectral temporal graph neural network for multivariate time-series forecasting," in *NeurIPS*, vol. 33, 2020, pp. 17 766–17 778.

[10] M. Jin, Y. Zheng, Y.-F. Li, S. Chen, B. Yang, and S. Pan, "Multivariate time series forecasting with dynamic graph neural odes," *IEEE TKDE*, 2022.

[11] L. Luo, G. Haffari, and S. Pan, "Graph sequential neural ode process for link prediction on dynamic and sparse graphs," in *WSDM*, 2023, pp. 778–786.

[12] M. Jin, Y.-F. Li, and S. Pan, "Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs," in *NeurIPS*, 2022.

[13] X. Wang and M. Zhang, "How powerful are spectral graph neural networks," in *ICML*, 2022.

[14] S. Kiefer, "Power and limits of the weisfeiler-leman algorithm," Ph.D. dissertation, Dissertation, RWTH Aachen University, 2020, 2020.

[15] H. Lütkepohl, "Vector autoregressive models," in *Handbook of Research Methods and Applications in Empirical Macroeconomics*. Edward Elgar Publishing, 2013, pp. 139–164.

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

[17] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *NeurIPS*, 2015, pp. 802–810.

[18] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint*, vol. abs/1803.01271, 2018.

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

[20] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," in *NeurIPS*, 2018, pp. 7796–7805.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008.

[22] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," in *IJCAI*, 2023.

[23] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *NeurIPS*, vol. 32, 2019.

[24] N. Kitaev, L. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," in *ICLR*, 2020.

[25] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *AAAI*, 2021, pp. 11 106–11 115.

[26] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *ICML*, vol. 162. PMLR, 2022, pp. 27 268–27 286.

[27] T. Zhou, Z. Ma, xue wang, Q. Wen, L. Sun, T. Yao, W. Yin, and R. Jin, "Film: Frequency improved legendre memory model for long-term time series forecasting," in *NeurIPS*, 2022.

[28] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *ICLR*, 2018.

[29] T. Sofianos, A. Sampieri, L. Franco, and F. Galasso, "Space-time-separable graph convolutional network for pose forecasting," in *CVPR*, 2021, pp. 11 209–11 218.

[30] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *AAAI*, 2021, pp. 4189–4196.

[31] Z. Fang, Q. Long, G. Song, and K. Xie, "Spatial-temporal graph ode networks for traffic flow forecasting," in *KDD*, 2021, pp. 364–373.

[32] A. Prabowo, W. Shao, H. Xue, P. Koniusz, and F. D. Salim, "Because every sensor is unique, so is every pair: Handling dynamicity in traffic forecasting," *arXiv preprint*, vol. abs/2302.09956, 2023.

[33] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *AAAI*, vol. 33, 2019, pp. 922–929.

[34] R. Huang, C. Huang, Y. Liu, G. Dai, and W. Kong, "Lsgcn: Long short-term traffic prediction with graph convolutional networks." in *IJCAI*, vol. 7, 2020, pp. 2355–2361.

[35] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *ICLR*, 2019.

[36] B. Xu, H. Shen, Q. Cao, K. Cen, and X. Cheng, "Graph convolutional networks using heat kernel for semi-supervised learning," in *IJCAI*, 2019, pp. 1928–1934.

[37] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[38] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Normalizing flow-based neural process for few-shot knowledge graph completion," in *SIGIR*, 2023.

[39] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Graph neural networks with convolutional arma filters," *IEEE TPAMI*, vol. 44, no. 7, pp. 3496–3507, 2021.

[40] S. Zheng, Z. Zhu, Z. Liu, Y. Li, and Y. Zhao, "Node-oriented spectral filtering for graph neural networks," *IEEE TPAMI*, 2023.

[41] Y. Wang, C. Xu, C. Xu, and D. Tao, "Packing convolutional neural networks in the frequency domain," *IEEE TPAMI*, vol. 41, no. 10, pp. 2495–2510, 2018.

[42] A. H. Souza, D. Mesquita, S. Kaski, and V. K. Garg, "Provably expressive temporal graph networks," in *NeurIPS*, 2022.

[43] J. M. Wallace and R. E. Dickinson, "Empirical orthogonal representation of time series in the frequency domain. part i: Theoretical considerations," *Journal of Applied Meteorology and Climatology*, vol. 11, no. 6, pp. 887–892, 1972.

[44] M. Poli, S. Massaroli, F. Berto, J. Park, T. Dao, C. Re, and S. Ermon, "Transform once: Efficient operator learning in frequency domain," in *NeurIPS*, 2022.

[45] A. Marco, J.-J. Martı *et al.*, "Polynomial least squares fitting in the bernstein basis," *Linear Algebra and its Applications*, vol. 433, no. 7, pp. 1254–1264, 2010.

[46] R. Sen, H.-F. Yu, and I. S. Dhillon, "Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting," in *NeurIPS*, vol. 32, 2019.

[47] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NeurIPS*, vol. 27, 2014.

[48] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock price prediction via discovering multi-frequency trading patterns," in *KDD*, 2017, pp. 2141–2149.

[49] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Q. Z. Sheng, "Stg2seq: spatial-temporal graph to sequence model for multi-step passenger demand forecasting," in *IJCAI*, 2019, pp. 1981–1987.

[50] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *NeurIPS*, vol. 34, 2021, pp. 22 419–22 430.

[51] J. Choi, H. Choi, J. Hwang, and N. Park, "Graph neural controlled differential equations for traffic forecasting," in *AAAI*, 2022, pp. 6367–6374.

[52] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[53] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[54] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, "Relative-error cur matrix decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 2, pp. 844–881, 2008.

APPENDIX A
PROOFS

A.1 Proof of Theorem 1

Theorem 1. *A linear* SPTGNN *can produce arbitrary dimensional time series representations at any valid time iff:* ***(1)*** $\hat{\mathbf{L}}$ *has no repeated eigenvalues;* ***(2)*** $\mathbf{X}$ *encompasses all frequency components with respect to the graph spectrum;* ***(3)*** $\mathcal{T}_\phi(\cdot)$ *can express any single-dimensional univariate time series.*

Proof. Let us assume that $t = 1$ (i.e., there is only one snapshot $\mathcal{G}_t$ of the graph $\{\mathcal{G}_t\}_{t=0}^{T-1}$), $\mathcal{T}_\phi(\cdot)$ reduces to linear functions of a graph snapshot and the linear spectral-temporal GNN reduces to a linear spectral GNN, which can be equivalently written as:

$$\mathbf{Z}_t = g_\theta(\hat{\mathbf{L}})\mathbf{X}_t\mathbf{\Phi} \in \mathbb{R}^{N\times D}. \tag{10}$$

We first prove the universality theorem of linear spectral GNNs in a graph snapshot on the basis of Theorem 4.1 in [13]. In other words, assuming $\tilde{\mathbf{X}}_t = \mathbf{U}^\top\mathbf{X}_t$ has non-zero row vectors and $\hat{\mathbf{L}}$ has unique eigenvalues, we first aim to prove that for any $\mathbf{Z}_{t,d} \in \mathbb{R}^{N\times 1}$, there is a linear spectral GNN to produce it. We assume there exists $\boldsymbol{\phi}^* \in \mathbb{R}^D$ s.t. all elements in $\tilde{\mathbf{X}}_t\boldsymbol{\phi}^*$ are non-zero. Considering a case where $(\tilde{\mathbf{X}}_t\boldsymbol{\phi})_i = 0$ and letting the solution space to be $\mathbb{S}_i$, we know that $\mathbb{S}_i$ is a proper subspace of $\mathbb{R}^D$ as the i-th row of $\tilde{\mathbf{X}}_t$ is non-zero. Therefore, $\mathbb{R}^D \setminus \cup_{i=1}^N \mathbb{S}_i \neq \emptyset$, and we know that all vectors $\boldsymbol{\phi}$ in $\mathbb{R}^D \setminus \cup_{i=1}^N \mathbb{S}_i$ are valid to form $\boldsymbol{\phi}^*$. We then filter $\tilde{\mathbf{X}}_t\boldsymbol{\phi}^*$ to get $\mathbf{Z}_{t,d}$. Firstly, we let $\tilde{\mathbf{Z}}_{t,d} = \mathbf{U}^\top\mathbf{Z}_{t,d}$ and assume there is a polynomial with $N-1$ order:

$$\begin{aligned} p_i &:= g_\theta(\lambda_i), \\ &= \sum_{k=0}^{N-1} \theta_k \lambda_i^k \quad \text{s.t.} \quad p_i = \tilde{\mathbf{Z}}_{t,d}[i]/(\tilde{\mathbf{X}}_t\boldsymbol{\phi}^*)_i \ , \ \forall i \in \{1,\cdots,N\}. \end{aligned} \tag{11}$$

On this basis, the polynomial coefficients $\boldsymbol{\theta}$ is the solution of a linear system $\mathbf{B}\boldsymbol{\theta} = \mathbf{p}$ where $\mathbf{B}_{i,j} = \lambda_i^{j-1}$. Since λ_i are different from each other, $\mathbf{B}^\top$ turns to a nonsingular Vandermonde matrix, where a solution $\boldsymbol{\theta}$ always exists. Therefore, a linear spectral GNN can produce any one-dimensional prediction under certain assumptions.

The above proof states that linear spectral GNNs can produce any one-dimensional prediction if $\hat{\mathbf{L}}$ has no repeated eigenvalues (i.e., condition 1) and the node features $\mathbf{X}$ contain all frequency components w.r.t. graph spectrum (i.e., condition 2). When $t \geq 2$, $\mathcal{T}_\phi(\cdot)$ turns to linear functions over all historical observations of graph snapshots. In order to distinguish between different historical graph snapshots, $\mathcal{T}_\phi(\cdot)$ must be universal approximations of all historical graph snapshots, implying that $\mathcal{T}_\phi(\cdot)$ is able to express any one-dimensional univariate time series (i.e., condition 3). □

A.2 Proof of Theorem 2

Theorem 2. *For a linear* SPTGNN *with valid temporal FDMs that can express arbitrary 1-dimensional univariate time series and a K-degree polynomial basis in its GSFs,* $\forall u, v \in \mathbb{V}, \mathbf{Z}_t[u] = \mathbf{Z}_t[v]$ *if* $\mathbf{C}^{(K+1)}(u,t) = \mathbf{C}^{(K+1)}(v,t)$. $\mathbf{Z}_t[i]$ *and* $\mathbf{C}^{(K)}(i,t)$ *represent node i's embedding at time t in such a GNN and the K-step temporal 1-WL test, respectively.*

Proof. Given valid temporal frequency-domain models (i.e., space projectors and TSFs) and a K-degree polynomial filter function, the prediction of a linear SPTGNN can be formulated as follows.

$$\mathbf{Z} = \mathcal{T}_\phi\left(\sum_{k=0}^{K} \mathbf{\Theta}_k P_k(\hat{\mathbf{L}})\mathbf{X}\right). \tag{12}$$

For ease of reading, we redefine $\mathcal{T}_\phi(\cdot)$ as the combination of space projections and TSFs in the following proof. Let us assume $t = 1$ (i.e., there is only one snapshot $\mathcal{G}_t$ of the graph $\{\mathcal{G}_t\}_{t=0}^{T-1}$), $\mathcal{T}_\phi(\cdot)$ reduces to linear functions of a single graph snapshot and a linear SPTGNN reduces to a linear spectral GNN. Using the framework in [36], Eq. 12 can be viewed as a $k+1$-layer GNN. The output of the last layer in GNN produces the output of linear spectral GNNs [13]. According to the proof of Lemma 2 in [36], if WL node labels $\mathbf{C}^{(K+1)}(u) = \mathbf{C}^{(K+1)}(v)$, the corresponding GNN's node features should be the same at any iteration. Therefore, for all nodes $\forall u, v \in \mathbb{V}, \mathbf{Z}[u] = \mathbf{Z}[v]$ if $\mathbf{C}^{(K+1)}(u) = \mathbf{C}^{(K+1)}(v)$.

When $t \geq 2$, we have a DTDG defined as a sequence of graph snapshots $(\mathcal{G}_1, \mathcal{G}_2, \ldots)$ that are sampled at regular intervals, and each snapshot is a static graph. Note that any DTDGs can be equivalently converted to continuous-time temporal graphs (CTDGs). The CTDG can be equivalently viewed as time-stamped multi-graphs with timestamped edges, i.e., $\mathcal{G}(t) = \{(u_k, v_k, t_k) \mid t_k < t\}$. According to Proposition 6 in [42], the expressive power of dynamic GNN with injective message passing is bounded by the temporal WL test on $\mathcal{G}(t)$. Since $\mathcal{T}_\phi(\cdot)$ is a set of linear functions over all historical observations of graph snapshots, i.e., $\mathcal{T}_\phi(\cdot)$ represents linear transformations of $\mathcal{G}(t)$. The defined SPTGNN, i.e., $\mathbf{Z} = \mathcal{T}_\phi(\sum_{k=0}^K \mathbf{\Theta}_k P_k(\hat{\mathbf{L}})\mathbf{X})$, should be as expressive as dynamic GNN with injective message passing. Hence, if temporal WL node labels $\mathbf{C}^{(K+1)}(u,t) = \mathbf{C}^{(K+1)}(v,t)$, the corresponding GNN's node features should be the same at any timestamp t and at any iteration. Therefore, for all nodes $\forall u, v \in \mathbb{V}, \mathbf{Z}_t[u] = \mathbf{Z}_t[v]$ if $\mathbf{C}^{(K+1)}(u,t) = \mathbf{C}^{(K+1)}(v,t)$. □

A.3 Proof of Proposition 2

Proposition 2. *If a discrete-time dynamic graph with a fixed graph topology at time t has no repeated eigenvalues in its normalized graph Laplacian and has no missing frequency components in each snapshot, then the temporal 1-WL is able to differentiate all non-isomorphic nodes at time t.*

Proof. Assume there are no repeated eigenvalues and missing frequency components w.r.t. graph spectrum in a DTDG with fixed topology and time-evolving features, i.e., $\{\mathcal{G}_t\}_{t=0}^{T-1}$.

According to Corollary 4.4 in [13], we know that if a graph has no repeated eigenvalues and missing frequency components, then 1-WL test can differentiate any pair of non-isomorphic nodes. We denote the colors of two nodes u and v in $\mathcal{G}_t$ after L 1-WL interactions as $\mathbf{C}^{(L)}(u,t)$ and $\mathbf{C}^{(L)}(v,t)$ s.t. $\mathbf{C}^{(L)}(u,t) \neq \mathbf{C}^{(L)}(v,t)$ if u and v are non-isomorphic. On this basis, we consider two scenarios in

$\mathcal{G}_{t+1}$: **(1)** two or more non-isomorphic nodes have identical initial colors; **(2)** none of the non-isomorphic nodes have identical colors. Under the assumptions in this proposition, the 1-WL test can differentiate u and v in $\mathcal{G}_{t+1}$ on both cases with different

$$\mathbf{C}^{(L)}(u,t+1) := \textsc{Hash}(c^{(L-1)}(u,t+1), \{\!\{c^{(L-1)}(m,t+1) : e_{u,m,t+1} \in \mathbb{E}(\mathcal{G}_{t+1})\}\!\})$$

and

$$\mathbf{C}^{(L)}(v,t+1) := \textsc{Hash}(c^{(L-1)}(v,t+1), \{\!\{c^{(L-1)}(m,t+1) : e_{v,m,t+1} \in \mathbb{E}(\mathcal{G}_{t+1})\}\!\}).$$

Therefore, no matter whether $\mathbf{C}^{(L)}(u,t)$ and $\mathbf{C}^{(L)}(v,t)$ are identical or not (they are different in fact as mentioned), we have nonidentical

$$\mathbf{C}^{(L)}(u,t+1) := \textsc{Hash}(c^{(L-1)}(u,t+1), c^{(L-1)}(u,t), \{\!\{c^{(L-1)}(m,t+1) : e_{u,m,t+1} \in \mathbb{E}(\mathcal{G}_{t+1})\}\!\})$$

and

$$\mathbf{C}^{(L)}(v,t+1) := \textsc{Hash}(c^{(L-1)}(v,t+1), c^{(L-1)}(v,t), \{\!\{c^{(L-1)}(m,t+1) : e_{v,m,t+1} \in \mathbb{E}(\mathcal{G}_{t+1})\}\!\})$$

in the temporal 1-WL test, where $\mathbf{C}^{(L)}(u,t) := c^{(L-1)}(u,t)$ and $\mathbf{C}^{(L)}(v,t) := c^{(L-1)}(v,t)$. □

A.4 Proof of Proposition 3

Proposition 3. *If a discrete-time dynamic graph with a fixed graph topology has no multiple eigenvalues in its normalized graph Laplacian and has no missing frequency components in each snapshot, then no automorphism exists.*

Proof. Given a DTDG $\{\mathcal{G}_t\}_{t=0}^{T-1}$ consists of T static graph snapshots with fixed graph topology and time-evolving node features, we first prove that all pairs of nodes are non-isomorphic.

In a snapshot $\mathcal{G}_t$, assume there is a permutation matrix $\mathbf{P}$, we have

$$\hat{\mathbf{L}} := \mathbf{P}^\top \hat{\mathbf{L}} \mathbf{P} = \mathbf{P}\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top\mathbf{P}^\top, \tag{13}$$

and we know

$$\mathbf{\Lambda} := \mathbf{U}^\top\mathbf{P}\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top\mathbf{P}^\top\mathbf{U} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top, \tag{14}$$

where $\mathbf{V}$ is an orthogonal matrix. Since all diagonal elements in $\mathbf{\Lambda}$ are different because we assume no repeated eigenvalues, then the eigenspace of each eigenvalue has only one dimension [13]; thus, we have $\mathbf{U}^\top\mathbf{P}\mathbf{U} = \mathbf{D}$, where $\mathbf{D}$ is a diagonal matrix s.t. $\mathbf{V} := \mathbf{D}$ with ± 1 elements. Now considering the node features $\mathbf{X}_t := \mathbf{P}\mathbf{X}_t$, we have $\hat{\mathbf{X}}_t = \mathbf{V}\hat{\mathbf{X}}_t$; thus $(\mathbf{I} - \mathbf{D})\hat{\mathbf{X}}_t = 0$ based on the above discussion. If there are no missing frequency components in $\hat{\mathbf{X}}_t$, i.e., no zero row vectors, we have $\mathbf{D} = \mathbf{I}$ and know that

$$\mathbf{P} = \mathbf{U}\mathbf{D}\mathbf{U}^\top = \mathbf{I}. \tag{15}$$

Hence, we prove that all nodes in a graph snapshot $\mathcal{G}_t$ are non-isomorphic. In $\{\mathcal{G}_t\}_{t=0}^{T-1}$, we have $\mathbf{V} := \mathbf{D}$ always holds across all snapshots if its normalized graph Laplacian has no repeated eigenvalues. On this basis, if there are no missing frequency components by giving $\mathbf{X}_t, \forall t \in \{0, 1, \ldots, T-1\}$, all pairs of nodes are non-isomorphic in an attributed DTDG with fixed graph topology.

□

A.5 Proof of Theorem 3

Theorem 3. *For a linear* SPTGNN *optimized with mean squared loss, any complete polynomial bases result in the same expressive power, but an orthonormal basis guarantees the maximum convergence rate if its weight function matches the graph signal density.*

Proof. Directly analyzing Eq. 6 is complex and unnecessary to study the effectiveness of different polynomial bases when learning time series relations at each time step. Since optimizing the spectral-temporal GNNs formulated in Eq. 6 can be understood as a two-step (i.e., graph-then-temporal) optimization problem, we directly analyze the optimization of $\mathbf{\Theta}$ w.r.t. the formulation below based on the squared loss $\mathcal{L} = \frac{1}{2}||\mathbf{Z}_t - \mathbf{Y}_t||_F^2$ on a graph snapshot $\mathcal{G}_t$ with the target $\mathbf{Y}_t$.

$$\mathbf{Z}_t = \sum_{k=0}^{K} \mathbf{\Theta}_k P_k(\hat{\mathbf{L}})\mathbf{X}_t. \tag{16}$$

This is a convex optimization problem, thus the convergence rate of gradient descent relates to the condition number of the Hessian matrix [53]. In other words, the convergence rate reaches the maximum if $\kappa(\mathbf{H})$ reaches the minimum. We have $\mathbf{H}_{k_1,k_2}$ defined as follows that is similar in [13].

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{\Theta}_{k_1} \partial \mathbf{\Theta}_{k_2}} &= \mathbf{X}_t^\top P_{k_2}(\hat{\mathbf{L}}) P_{k_1}(\hat{\mathbf{L}}) \mathbf{X}_t, \\ &= \sum_{i=1}^{n} P_{k_2}(\lambda_i) P_{k_1}(\lambda_i) \tilde{\mathbf{X}}_t[\lambda_i].\end{aligned} \tag{17}$$

This equation can be written as a Riemann sum as follows.

$$\frac{\partial \mathcal{L}}{\partial \mathbf{\Theta}_{k_1} \partial \mathbf{\Theta}_{k_2}} = \sum_{i=1}^{n} P_{k_2}(\lambda_i) P_{k_1}(\lambda_i) \frac{F(\lambda_i) - F(\lambda_{i-1})}{\lambda_i - \lambda_{i-1}} (\lambda_i - \lambda_{i-1}). \tag{18}$$

In the above formula, $F(\lambda_i) := \sum_{\lambda_j \le \lambda_i} (\tilde{\mathbf{X}}_t[\lambda_j])^2$ and $\frac{F(\lambda_i) - F(\lambda_{i-1})}{\lambda_i - \lambda_{i-1}}$ denotes the graph signal density at the frequency λ_i. When $n \to \infty$, we have the (k_1, k_2) element in $\mathbf{H}$ rewrite as follows.

$$\mathbf{H}_{k_1,k_2} = \int_{\lambda=0}^{2} P_{k_2}(\lambda) P_{k_1}(\lambda) \frac{\Delta F(\lambda)}{\Delta \lambda} \, d\lambda. \tag{19}$$

We know that $\kappa(\mathbf{H})$ reaches the minimum if $\mathbf{H}$ is a diagonal matrix, which tells that the polynomial bases, e.g., $P_{k_1}(\lambda)$ and $P_{k_2}(\lambda)$, should be orthogonal w.r.t. the weight function $\frac{\Delta F(\lambda)}{\Delta \lambda}$.

□

A.6 Proof of Lemma 2

Lemma 2. *Suppose the projection of* $\mathbf{A}$ *by* $\mathbf{A}'$ *is* $P_{\mathbf{A}'}(\mathbf{A})$*, and the coherence measure of* $\mathbf{A}$ *is* $\mu(\mathbf{A}) = \Omega(k/N)$*, then with a high probability, the error between* $\mathbf{AW}$ *and* $P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}$ *is bounded by* $||\mathbf{AW} - P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}||_F \leq (1+\epsilon)||\mathbf{W}||_F||\mathbf{A} - \mathbf{A}_k||_F$ *if* $S = O(k^2/\epsilon^2)$.

Proof. Similar to the analysis in Theorem 3 from [54] and Theorem 1 from [26], we have

$$\begin{aligned} ||\mathbf{AW} - P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}||_F &\leq ||\mathbf{W}||_F||\mathbf{A} - P_{\mathbf{A}'}(\mathbf{A})||_F \\ &= ||\mathbf{W}||_F||\mathbf{A} - \mathbf{A}'(\mathbf{A}')^\dagger\mathbf{A}||_F \\ &\leq ||\mathbf{W}||_F||\mathbf{A} - (\mathbf{AS}^\top)(\mathbf{A}_k\mathbf{S}^\top)^\dagger\mathbf{A}_k||_F. \end{aligned} \tag{20}$$

Then, following Theorem 5 from [54], if $S = O(k^2/\epsilon^2 \times \mu(\mathbf{A})N/k)$, we can obtain the following result with a probability at least 0.7

$$||\mathbf{A} - (\mathbf{AS}^\top)(\mathbf{A}_k\mathbf{S}^\top)^\dagger\mathbf{A}_k||_F \leq (1+\epsilon)||\mathbf{A} - \mathbf{A}_k||_F. \tag{21}$$

Since $\mu(\mathbf{A}) = \Omega(k/N)$, when $S = O(k^2/\epsilon^2)$ together with Eq. 20 and Eq. 21, we can obtain the final bound as

$$||\mathbf{AW} - P_{\mathbf{A}'}(\mathbf{A})\mathbf{W}||_F \leq (1+\epsilon)||\mathbf{W}||_F||\mathbf{A} - \mathbf{A}_k||_F. \tag{22}$$

□

APPENDIX B
ADDITIONAL EXPERIMENTAL SETTINGS

The detailed hyperparameter configurations are in Tab. 7 and Tab. 8.

TABLE 7: The hyperparameter setting of our method for short-term time series forecasting.

Hyperparameters	PeMS03	PeMS04	PeMS07	PeMS08	Electricity	Solar	ECG
Gegenbauer parameter α	3.08	0.47	1	1	1.2	1.2	1.2
# polynomial degree K	4	4	4	4	4	4	4
# selected component S	5	4	5	5	5	5	5
# building block M	2	2	2	2	2	2	2
Training batch size B	32	64	50	50	50	50	50
Model learning rate η	0.0003	0.001	0.001	0.001	0.001	0.001	0.001

TABLE 8: The hyperparameter setting of our method for long-term time series forecasting.

Hyperparameters	α	β	K	S	M	B	η
Electricity	1	1	4	14	5	50	0.001
Solar-Energy	1	1	4	50	5	50	0.001
Weather	0.81	0.90	4	26	3	64	0.0003

APPENDIX C
ADDITIONAL RESULTS

C.1 Additional Forecasting Results

We provide our supplementary short-term forecasting results in Tab. 9, from which the following observations can be made: **(1)** TGGC, in most cases, outperforms all baseline methods by substantial margins, with an average improvement of 35% compared to the best deep time series baselines; **(2)** it generally outperforms StemGNN, although the performance gaps are not very significant under this experimental setting.

TABLE 9: Additional short-term forecasting results on three time series benchmarks, where we follow [9] for the experimental setting and baseline results. We use the **bold** and underline fonts to indicate the best and second-best results.

Method	MAE	RMSE	MAE	RMSE	MAE	RMSE
	Electricity		*Solar*		*ECG*	
FC-LSTM [47]	0.62	0.20	0.13	0.19	0.32	0.54
TCN [18]	0.07	0.51	0.06	0.06	0.10	0.30
LSTNet [19]	0.06	0.07	0.07	0.19	0.08	0.12
DeepState [20]	0.06	0.67	0.06	0.25	0.09	0.76
DeepGLO [46]	0.08	0.14	0.09	0.14	0.09	0.15
SFM [48]	0.08	0.13	0.05	0.09	0.17	0.58
GWNet [3]	0.07	0.53	0.09	0.14	0.09	0.15
StemGNN [9]	**0.04**	**0.06**	0.03	0.07	0.05	0.07
TGGC **(Ours)**	0.05	0.07	**0.02**	**0.04**	**0.04**	**0.06**

C.2 Main Result Statistics

Tab. 10 and Tab. 11 present the average performances and 95% confidence intervals for our results reported in Tab. 2, Tab. 3, and Tab. 5 with 5 individual runs.

TABLE 10: Short-term forecasting results showing our average performances $\pm 95\%$ confidence intervals, respectively.

Method		TGGC		TGGC†	
		MAE	RMSE	MAE	RMSE
PeMS03	3	13.52 ± 0.226	21.74 ± 0.687	-	-
	12	-	-	16.22 ± 0.475	27.07 ± 0.623
PeMS04	3	18.77 ± 0.142	29.92 ± 0.150	-	-
	12	-	-	20.00 ± 0.300	32.10 ± 0.452
PeMS07	3	1.92 ± 0.029	3.35 ± 0.093	-	-
	12	-	-	2.81 ± 0.035	5.58 ± 0.066
PeMS08	3	14.55 ± 0.245	22.73 ± 0.274	-	-
	12	-	-	16.54 ± 0.981	26.10 ± 0.825

TABLE 11: Long-term forecasting results showing our average performances $\pm 95\%$ confidence intervals, respectively.

Method		TGGC†	
		MAE	RMSE
Electricity	96	0.293 ± 0.004	0.425 ± 0.006
	192	0.303 ± 0.006	0.440 ± 0.007
	336	0.313 ± 0.007	0.470 ± 0.005
Weather	96	0.235 ± 0.014	0.408 ± 0.029
	192	0.286 ± 0.021	0.468 ± 0.041
	336	0.317 ± 0.019	0.515 ± 0.036
Solar	96	0.242 ± 0.005	0.443 ± 0.009
	192	0.263 ± 0.004	0.470 ± 0.010
	336	0.271 ± 0.004	0.478 ± 0.009