Abstract—The adaptive estimation of coexisting temporal vertex (node) and edge signals on graphs is a critical task when a change in edge signals influences the temporal dynamics of the vertex signals. However, the current Graph Signal Processing algorithms mostly consider only the signals existing on the graph vertices and have neglected the fact that signals can reside on the edges. We propose an Adaptive Joint Vertex-Edge Estimation (AJVEE) algorithm for jointly estimating time-varying vertex and edge signals through a time-varying regression, incorporating both vertex signal filtering and edge signal filtering. Accompanying AJVEE is a newly proposed Adaptive Least Mean Square procedure based on the Hodge Laplacian (ALMS-Hodge), which is inspired by classical adaptive filters combining simplicial filtering and simplicial regression. AJVEE is able to operate jointly on the vertices and edges by merging two ALMS-Hodge specified on the vertices and edges into a unified formulation. A more generalized case extending AJVEE beyond the vertices and edges is being discussed. Experimenting on real-world traffic networks and population mobility networks, we have confirmed that our proposed AJVEE algorithm could accurately and jointly track time-varying vertex and edge signals on graphs.

Index Terms—graph signal processing (GSP), time series, simplicial complexes, adaptive algorithms, and graph learning.

I. INTRODUCTION

Graphs have recently become a popular and impactful research topic due to their effective ability to represent interacting multivariate signals. However, several challenges must be addressed to enhance the effectiveness of signal processing algorithms on multivariate time-varying signals that reside on graph-like topological structures. These challenges include representing topological irregularity, estimating unknown data from known data, removing noise present in the data, and extracting cross-space-time variations. In graph signal processing (GSP), graphs have demonstrated their capacity to represent information in various real-world scenarios and disciplines of studies. Graphs have been studied extensively and attracted considerable attention due to their irregular structure, as well as their effectiveness in representing interactions among data [1], [2], [3]. Measurements taken by multiple sensors could be used to construct sensor graphs based on the locations of the sensors, with the data features embedded on the vertices. Examples of sensor graphs include recordings of 5G signal reception strength graph [4], air-quality graph [5], and nationwide temperature graph [6]. In bioinformatics, biological networks can model time-varying gene interactions. [7], functional brain imaging [8], and biomedicine [9]. Graphical models are proposed in finance to reflect the interplay of the stock market from the interaction among stocks [10].

Despite the success of GSP, GSP solely considers data on the graph vertices, meaning that higher-order signals, such as signals on graph edges, are not being represented. To provide a few examples, traffic data on road networks recorded on the edges [11], [12], gene interaction over time forms a set of time-varying graph edges [7], multi-agent systems use time-varying edge data to model the agent interactions [13]. A citation complex can be viewed as a higher-ordered social network and formed from constructing simplices among co-author collaborations: the number of citations of a 2-author is a signal on the graph edges and the number of citations of a 3-author paper is a signal on a triangle [14]. The spread of COVID-19 across multiple regions can be represented using a series of dynamic graphs, where the vertices record the confirmed cases in each region, and the edges represent population mobility between regions. In existing GSP algorithms that process time-varying vertex signals, the interaction among vertices is modeled using fixed parameters at all time instances [15]. Failing to capture the time-varying interaction would lead to an inaccurate estimation.

The recently emerged Topological Signal Processing (TSP) has been established as an alternative point of view from GSP for higher-ordered signals, overcoming the limitation of GSP by generalizing graphs as simplicial complexes. The signals on the vertices of a graph are considered as signals of the 0 order simplex, while higher order simplices could be used to represent signals on the edges and beyond [16], [17]. In [11], semi-supervised learning for data on the graph edges is done using the Hodge Laplacians and flow conservation. An analogy of spectral wavelets on simplicial complexes called Hodgelets is proposed in [18]. The work of [19] further proposed topological Slepians, which are the signals concentrated on the topological domain and localized on the frequency domain. In [20], [21], finite impulse response filters on simplicial complexes were discussed, and the work of [22] further defined the convolution operation for signals on a simplicial complex. Recently, the proposal of using a Dirac operator allows the merging of fixed time-invariant signals from different orders of simplex [23]. Chebyshev polynomial approximations of spectral filters on simplicial complexes are provided in [22] and [24]. The sampling of graph edges using
the line graph transformation is discussed in [25].

The above TSP methods demonstrated good performance at processing signals on graph edges, but we may encounter additional challenges in realistic situations. Signals, in reality, often contain information about time dimension; when processing data on graph vertices that evolve with time, one could rely on graph adaptive filters, a combination of classical adaptive filters with graph shift operations. In GSP, the graph adaptive least mean squares (GLMS) algorithm was proposed for online estimation of vertex signals through $l_2$-norm optimization, assuming Gaussian noise and bandlimitedness of signal [4]. Extensions of GLMS have improved its performance in various aspects. For instance, the normalized GLMS (GNLMS) incorporates spectral domain normalization in the bandlimited filter, enhancing convergence speed compared to GLMS [6]. Other algorithms such as the Graph Least Mean $p$th (GLMP) [26], normalized GLMP [27], and Graph Alternating Minimization (GAM) algorithms have increased robustness against impulsive noise. The adaptive Graph-Sign (G-Sign) reduces computational complexity and increases robustness by employing an update function based on $l_1$ norm optimization [29], [30].

Another class of algorithms that processes time-varying vertex signals combines time-series analysis techniques such as Vector Autoregressive (VAR), Vector Autoregressive Moving Average (VARMA), and generalized autoregressive conditionally heteroscedastic (GARCH) models GSP, resulting in the graph VAR [31], graph VARMA [32], [33], and graph GARCH [15]. Recently, the discussions of time-varying signals beyond the graph vertices have emerged: online water flow imputation on graph edges was attempted strictly using flow conservation in [34]; the simplicial VAR was proposed in [35] on the same water flow imputation task. The major drawback of implementing time-series analysis techniques on topological structures is that the multivariate nature of topological signals increases model complexity and poses challenges in acquiring accurate models. Research on the online estimation of signals on simplicial complexes is limited, particularly for the joint task of spatial signal imputation and temporal adaptive predictions.

Prior TSP and GSP methods have another limitation of primarily focusing on signals in a specific simplicial order. In other words, the task to be solved is only on the vertices or the edges, but rarely both, not to mention that the majority of previous GSP and TSP methods are targeted for time-invariant scenarios. The need to process signals residing in multiple orders of simplices has drawn increased attention to the interactions and appearances of signals across these different simplicial orders. In biological networks, transportation networks, and population mobility networks examples, the graph vertices and edges both contain signals, with the edge signals influencing the behavior of the vertex signals [2], [13], [36]. Understanding the dynamic interactions within these networks and signals from different simplicial orders is a crucial task for facilitating online tracking and predictive decision-making algorithms to enhance efficiency and reliability in real-world applications. On a side note, data gathered from the real world is often noisy, meaning that the denoising of simplicial data should be considered when formulating simplicial representations. These challenges comprise addressing topological irregularities, revealing data interactions across multiple dimensions, eliminating data noise, and capturing time-varying patterns. Therefore, it is a pressing necessity for an online algorithm that focuses on time-varying multi-order and multi-variate data on simplicial complexes to bridge the cross-space-time signal gaps.

In this paper, we propose the Adaptive Joint Vertex-Edge Estimation (AJVEE) algorithm as a strategy aimed at addressing the previously mentioned focuses for time-varying signals existing on both the vertices and the edges by considering the dynamic interplay between the signals on the vertices and edges. To the best of our knowledge, this is the first paper to consider the joint online estimation of time-varying signals on both the vertices and the edges. The major contributions of this paper are listed as follows:

- The AJVEE framework is proposed for adaptive estimation of time-varying vertex and edge signals on graphs by integrating two separate online estimations for vertices and edges and then into a single time-varying regression model. AJVEE utilizes a time-varying regression model to capture the interaction between vertex signals, with the intensity of these interactions represented by edge signals. AJVEE can be further generalized to operate beyond just vertices and edges, allowing for more complex signal interactions.

- An adaptive Least Mean Square procedure based on the Hodge Laplacian (ALMS-Hodge) is derived to support AJVEE. Unlike previous adaptive GSP algorithms that only address vertex estimation errors, ALMS-Hodge is generalized to higher-order structures such as edges and triangles. ALMS-Hodge relies upon both aggregated signals and estimation errors to conduct missing signal imputation in the spatial domain and adaptive predictions in the temporal domain.

In Section II we will provide preliminary knowledge in representing graphs using simplicial complexes, as well as basic signal processing techniques on simplicial complexes. The ALMS-Hodge is introduced in Section III. Derivations and analysis of the AJVEE are included in Section IV; we also provide a more generalized formulation of AJVEE applied to time-varying signals on simplicial complexes that are not restricted simply to the vertices and edges. Experiment results on both synthetic data and real data are discussed in Section V. Section VII concludes the paper with a brief discussion of potential future work. Table I summarizes some frequent symbols seen in this paper.

II. PRELIMINARY KNOWLEDGE

A. Graph representation

Let us begin with a vertex (node) set $V = \{v_1...v_N\}$. We will use the terms node and vertex interchangeably throughout the paper. A $k$-simplex $\Delta_k$ is defined as a subset of $V$ that has cardinality $k + 1$. A simplicial complex $K$ of order $K$ can be defined as a collection of $k$-simplices $\{\Delta_{k_1}...\Delta_{k_N}\}$, with $k = 0...K$. A simplicial complex $K$ has the property that for a simplex $\Delta_k \in K$, if another simplex with a lower order $\Delta_{k-1}$ is a subset of $\Delta_k$, then $\Delta_{k-1} \in K$ [16]. Following
### Table I: Symbols and their meanings

| Symbol | Meaning |
|--------|---------|
| $k$    | Order of a simplex or simplicial complex |
| $\mathcal{K}$ | $k$-simplex (vertex: $k = 0$; edge: $k = 1$) |
| $\mathcal{G}$ | simplicial complex graph |
| $t$    | time index |
| $x_{k}[t] \in \mathbb{R}^{N_k}$ | $k$-simplicial signal of time $t$ |
| $N_k$  | Cardinality of $x_{k}[t]$ |
| $L_k \in \mathbb{R}^{N_k \times N_k}$ | Hodge Laplacian matrix |
| $D_{k,s} \in \mathbb{R}^{N_k \times N_k}$ | Sampling (Masking) Matrix |
| $U_k \in \mathbb{R}^{N_k \times N_k}$ | Eigenvector matrix of $L_k$ |
| $f_k$  | The frequency set of the bandlimited filter |
| $U_{k,f} \in \mathbb{R}^{N_k \times |F_k|}$ | Bandlimit eigenvector matrix of $L_k$ |
| $B_k \in \mathbb{R}^{N_{k-1} \times N_k}$ | Incidence matrix |
| $I$    | Identity matrix |
| $\Sigma_k \in \mathbb{R}^{N_k \times N_k}$ | Simplicial spectral filter |
| $\eta_k[t] \in \mathbb{R}^{N_k}$ | Estimated $k$-simplicial signal of time $t$ |
| $\eta_k[t] \in \mathbb{R}^{N_k}$ | SFT of $x_{k}[t]$ |
| $\gamma_k[t] \in \mathbb{R}^{N_k}$ | Noise of the $k$-simplicial signal |
| $\mu_k[t] \in \mathbb{R}^{N_k}$ | Noisy observation of $x_{k}[t]$ |
| $\tau_k$ | Aggregation weight |

With order $K$, the Hodge Laplacians can be defined as follows:

$$L_k = \begin{cases} B_1 B_k^T, & k = 0, \\ B_k^T B_k + B_{k+1}^T B_{k+1} & < k < K, \\ B_K^T B_K, & k = K. \end{cases}$$

The Hodge Laplacian $L_k$ can be split into lower Hodge Laplacian $L_{k,l} = B_k^T B_k$ and upper Hodge Laplacian $L_{k,u} = B_k B_{k+1}^T$, contributing from the lower-adjacency and the upper-adjacency of $k$-simplices in a simplicial complex [22].

The TSP analog of GFT is the Simplicial Fourier Transform (SFT) based on the spectral decomposition of the Hodge Laplacian matrix [16]:

$$L_k = U_k \Lambda_k U_k^T.$$  

(2)

The eigenvalue matrix is $\Lambda_k = \text{diag}(\lambda_{k,1}, ..., \lambda_{k,N_k})$ with the eigenvalues $\lambda_{k,i}$ sorted in increasing order and considered as the frequencies. Smaller eigenvalues are considered as lower frequencies and larger eigenvalues are considered as higher frequencies. Each eigenvalue in $\Lambda_k$ has a corresponding eigenvector in the eigenvector matrix $U_k = \{u_{k,1}, ..., u_{k,N_k}\}$. The forward SFT $s_k = U_k^T x_k$ of a spatial domain signal $x_k$ transforms it to the frequency domain signal $s_k$. The inverse SFT is $x_k = U_k s_k$ transforms the frequency domain signal back to the spatial domain. It is worth mentioning that the GFT is a special case of the SFT for signals defined only on the graph vertices [16]. The Hodge Laplacian is known as the graph Laplacian when $k = 0$ and the graph Helmholtzian when $k = 1$ [38]. To process the signals residing on the edges, we can use SFT with $k = 1$.

With the SFT defined, we can apply predefined simplicial filters $\Sigma$ to each frequency component in $L_k$ to conduct spectral domain operations on simplicial complexes. A simple bandlimited filter in $\lambda_k$ based on a frequency set $F_k$ is constructed by a diagonal matrix $\Sigma_{F,k}$, where a 1 on the $i^{th}$ diagonal element indicates that $\lambda_i \in F_k$ and 0 otherwise [16]. We can define several spectral TSP filters and linearly combine them to obtain the desired frequency response. The following procedure is a simple yet complete bandlimited spectral filtering of $x_k$:

$$x_k' = U_k \Sigma_{F,k} U_k^T x_k.$$  

(3)

If the signal $x_k$ is bandlimited with frequencies $F_k$, then $x_k = U_k \Sigma_{F,k} U_k^T x_k$. To simplify notation and make use of the sparsity gained from eliminating some frequency components, we will set $U_{k,F} = \text{support}(U_k \Sigma_{F,k})$, where support() is the operation of dropping the column with all zero elements. Notice that the filter $\Sigma_{F,k}$ is idempotent and self-adjoint, so with the simplified notation we have

$$U_{k,F} U_{k,F}^T = U_{k,F} \Sigma_{F,k} U_{k,F}^T$$  

(4)

and

$$U_{k,F}^T U_{k,F} = \textbf{I}.$$  

(5)

The filtering operation in (3) can be done using a spectral approach or a spatial approach [24]. To achieve similar effects as (3) in the spatial domain, a predefined bandlimited filter is approximated using a series of shifted Chebyshev polynomials,
and the spatial relationship is represented by the Hodge Laplacian. Assuming that spatial operations are conducted in the \( k \)-simplicies of simplicial complex \( K \), the shifted Chebyshev polynomial \( T_p(L_k) \) with order \( P \) is defined as

\[
T_p(L_k) = \begin{cases} 
1, & \text{if } p = 0, \\
\frac{2L_k - \lambda_{\text{max}}}{\lambda_{\text{max}}}, & \text{if } p = 1,
\frac{4L_k^2 - 2\lambda_{\text{max}}T_{p-1}(L_k) - T_{p-2}(L_k)}{\lambda_{\text{max}}} , & \text{if } p \geq 2.
\end{cases}
\]

Using (6), the approximation of \( U_k \Sigma_{F,k} U_k^T \) in (8) is expressed as

\[
U_k \Sigma_{F,k} U_k^T \approx \theta_0 + \sum_{p=1}^{P} \theta_p T_p(L_k),
\]

where \( \theta_p \) is the coefficient of the Chebyshev polynomial.

A special case of the bandlimited filter is the ideal low-pass filter, which is based on the (spatial) smoothness assumption of the signal [25]. The ideal low-pass filter \( \Sigma_{k,F_p} \) with cutoff frequency \( \lambda_{k,c} \) can be defined using the frequency set \( F_p \) and can be expressed as

\[
\Sigma_{k,F_p} = \text{diag}(\sigma_{k,F_p}),
\]

where \( \lambda_{k,i} \in F_p \) if \( i \leq c \), \( \lambda_{k,i} \notin F_p \) if \( i > c \), and \( 1 < c < N_k \) is the index of the cutoff frequency \( \lambda_{k,c} \). This means that the value assignment of the \( i \)-th element of \( \sigma_{k,F_p} \) follows \( \sigma_{k,F_p,i} = 1 \) if \( i \leq c \). According to the TSP sampling theory for signals defined on \( K \), partial observation is modeled using a sampling operation on \( D_k,S \) [16]. The sampling set \( S_k \) defines the diagonal matrix masking operation, where the sampling matrix \( D_{k,S} = \text{diag}(d_{k,1},\ldots,d_{k,N_k}) \), where \( d_{k,i} = 1 \) if \( x_{k,i} \in S_k \) and \( 0 \) otherwise [16]. The sampling matrix \( D_{k,S} \) is a diagonal matrix that is both idempotent and self-adjoint. When signal observations are missing due to various conditions or criteria other than by sampling, the matrix \( D_{k,S} \) still denotes the observation mask.

Before proceeding to AJVEE, it is necessary to discuss how to reconstruct a static signal \( x_k \) when only partial observation of the signal is obtainable. With the assumption that \( x_k \) is a bandlimited signal with frequencies \( F_k \), a reconstruction of \( x_k \) is possible if the conditions \( \|D_{k,S} U_k F_k^T D_{k,S}\|_2 < 1 \) and \( |F_k| \leq |S_k| \) are satisfied, where \( D_{k,S} = I - D_{k,S} \) [16].

III. THE ALMS-HODGE

Before discussing the AJVEE framework, we would like to introduce a novel adaptive filtering algorithm named the ALMS-Hodge for online time-varying simplicial signal prediction on the \( k \)-simplices. Let us consider a graph \( G \) as a simplicial complex \( K \) containing time-varying signal \( x_k[t] \). In classical signal processing, a simple data model can be used to represent the time-varying dynamics of signals:

\[
\dot{x}_k[t + 1] = x_k[t] + \Delta_k[t],
\]

where \( \Delta_k[t] \) is the change that lead \( x_k[t] \) to \( x_k[t+1] \). Suppose we have another signal \( y_k[t] \), the noisy observation of \( x_k[t] \) that has missing elements. This signal \( y_k[t] \) is modeled using a sampling operation on \( D_{k,S} \) and an i.i.d. additive Gaussian noise \( \eta_k[t] \) similar to what is found in GSP [4]:

\[
y_k[t] = D_k,S(x_k[t] + \eta_k[t]).
\]

We could set up a cost function \( J(\hat{x}_k[t]) \) that minimizes the squared error between \( y_k[t] \) and \( x_k[t] \) due to the fact that an \( l_2 \)-norm optimization problem yields the optimal solution under Gaussian noise assumption [39]:

\[
J(\hat{x}_k[t]) = f(\hat{x}_k[t]) + g(\hat{x}_k[t]),
\]

where \( f(\hat{x}_k[t]) = \mathbb{E} \|y_k[t] - D_{k,S} U_k F_k^T D_k,S \hat{x}_k[t]\|_2^2 \) is the \( l_2 \) cost that we are aiming to optimize, \( \hat{x}_k[t] \) is the estimation of the ground truth signal \( x_k[t] \), and \( g(\hat{x}_k[t]) \) is a task-specific regularization in terms of simplicial aggregation that we will be deriving later. At \( t = 0 \), the estimation \( x_k[0] \) is the initialization of the ALMS-Hodge. Here we assume that both the estimated signal \( \hat{x}_k[t] \) and \( x_k[t] \) are bandlimited and share the same frequency component. This consistency is ensured because both \( x_k[t] \) and \( \hat{x}_k[t] \) use the same topology, meaning that the SFT and frequency components are defined by the same Hodge Laplacian. The filter design of \( \Sigma_{k,F} \) can follow a universal filter such as the smoothness-based low-pass filter [25]. Another option we can choose is to use the SFT to acquire the spectrum directly from the historical data or estimate the spectrum from the noisy observation. We formulate (11) as a convex least mean squares optimization problem similar to GSP on the vertices [4]:

\[
\min_{x_k[t]} J(\hat{x}_k[t]), \text{ s.t. } U_k F_k^T D_k,S \hat{x}_k[t] = \hat{x}_k[t].
\]

The solution of (12) could be found by stochastic gradient.

In ALMS-Hodge, an essential step in online estimation is to effectively represent the change \( \Delta_k[t] \) on each simplex using TSP techniques. Looking at (12), we can represent \( \Delta_k[t] \) as the change between two-time points, which is obtained by the (negative) gradient of \( J(\hat{x}_k[t]) \). By letting \( \Delta_k[t] = -\frac{\partial J(\hat{x}_k[t])}{\partial \hat{x}_k[t]} \) in (9), an adaptive update function based on (9) for online simplicial signal estimation on order \( k \) using the Hodge Laplacian can be derived as

\[
\dot{x}_k[t + 1] = \dot{x}_k[t] + \Delta_k[t] = \dot{x}_k[t] - \frac{\partial J(\hat{x}_k[t])}{\partial \hat{x}_k[t]}.
\]

Combining bandlimitness of \( \dot{x}_k[t] \) and the idempotent property of \( D_{k,S} \), the term \( y_k[t] - D_{k,S} U_k F_k^T D_k,S \hat{x}_k[t] \) can be written as \( D_{k,S} (y_k[t] - U_k F_k^T D_k,S \hat{x}_k[t]) \). The update function of our ALMS-Hodge is derived as

\[
\dot{x}_k[t + 1] = \dot{x}_k[t] + \mu_k H_k e_k[t] + r_k R_k \hat{x}_k[t],
\]

where \( -2H_k e_k[t] = \frac{\partial J(\hat{x}_k[t])}{\partial \hat{x}_k[t]} \) and \( R_k \hat{x}_k[t] = \frac{\partial g(\hat{x}_k[t])}{\partial \hat{x}_k[t]} \). The term \( e_k[t] = D_{k,S} (y_k[t] - \hat{x}_k[t]) \) in (14) is the estimation error of all sampled part of the signals. Here, we can convert the spectral update into a general formulation of spatial update and use again the Chebyshev approximation in (7) to obtain a spatial update:

\[
H_k = U_k F_k^T D_k,S \approx \theta_0 + \sum_{p=1}^{P} \theta_p T_p(L_k).
\]
Step size parameters $\mu_k$ and $r_k$ are assigned to control the magnitude of the update terms as conventionally seen in classical adaptive filters [40]. The frequencies are defined as larger eigenvalues corresponding to high frequencies, so we can define $\Sigma_k$ as low-pass, high-pass, or band-limited filters similar to how these types of filters are designed in classical signal processing. Viewed in the spectral domain, each update term $\nu_k \mathbf{H}_k e_k[t]$ is applying the filter $\Sigma_k$ to the SFT of the error component $e_k[t]$. As a result, at each time instance $t$, the term $\mathbf{H}_k e_k[t]$ in (14) ensures the update of the ALMS-Hodge is time-varying and is adaptively generated in the direction opposite to the error. It can be shown that the ALMS-Hodge in (14) converges, for the details please check Appendix A.

If the term $r_k \mathbf{R}_k \hat{x}_k[t]$ is ignored as seen in previous adaptive GSP algorithms, the ALMS-Hodge will update solely based on the errors $e_k[t]$ but not utilizing the signals $x_k[t]$ itself. Thus, it is advantageous if we define aggregation terms based on the underlying signal to enforce additional restrictions or enhancements. For example, if we want to emphasize the property the harmonic component of a simplicial signal cannot be induced from the SFT of the signal of interest, we can still gain some insights by looking at them in the spectral domain. Generally speaking, by the Hodge decomposition (18), the nonzero eigenvalue-eigenvector pairs in $\mathbf{B}_k \mathbf{B}_k^T$ are orthogonal to the nonzero eigenvalue-eigenvector pairs in $\mathbf{B}_{k+1}^T \mathbf{B}_{k+1}$. All the nonzero eigenvalue-eigenvector pairs in $\mathbf{L}_k$ are either from $\mathbf{B}_k \mathbf{B}_k^T$ or $\mathbf{B}_{k+1}^T \mathbf{B}_{k+1}$ [16]. Another interesting property is $\mathbf{B}_k \mathbf{B}_{k+1} = 0$ because in the Hodge decomposition (18), the harmonic component of a simplicial signal cannot be induced from $\chi_{k+1}$ or $\chi_{k-1}$. From the orthogonality property, the harmonic components are associated with the zero eigenvalue-eigenvector pairs of $\mathbf{L}_k$.

There are three main advantages of the ALMS-Hodge. First, unlike the classical adaptive filter that learns the filters while making online estimation, our proposed ALMS-Hodge relies on assumptions such as spatial smoothness and bandlimitedness of the simplicial signal, then redefine universal TSP filters such as the low-pass filter. This means that the ALMS-Hodge will not be restricted to a specific order of simplicial signals. Additionally, not needing to learn the filters from the data reduces the complexity of the algorithm and simplifies the implementation in practice. Lastly, the ALMS-Hodge directly uses topological information of the data, which allows the missing simplicial signals to be estimated along with the temporal prediction. This is a unique characteristic of GSP and TSP algorithms that is not seen in classical signal processing.

IV. ADAPTIVE JOINT VERTEX-EDGE ESTIMATION

In this section, the AJVEE is proposed to target the case that the time-varying edge signal influences the time-varying vertex signal, from a simplicial diffusion perspective on a graph. The ALMS-Hodge will be used to facilitate the operation of AJVEE.
A. Vertex regression influenced by time-varying edge signals

In AJVEE, a TSP approach is taken to represent data interactions occurring between two simplices of order \( k = 1 \) and \( k = 0 \), corresponding to the edges and the vertices respectively. A time-varying vertex regression with the regression parameters represented in the form of the time-varying edge signal is modeled in AJVEE instead of using fixed edge weights as seen in GSP. To begin, we will formulate a time-varying regression of the vertices by representing the regression parameters using edge time-varying signals:

\[
x_{0}[t + 1] = \hat{L}_0[t]x_{0}[t] = B_1 \text{diag}(|x_{1}[t]|)B_1^T x_{0}[t],
\]

where \( \hat{L}_0[t] \) is a time-varying regression matrix derived from a graph topology containing a temporal edge signal, and \( t \) is the time index. We should emphasize that this process is fundamentally different from spatial node aggregation using purely the graph Laplacian \( L_0 \). The regression matrix \( \hat{L}_0[t] \) is formed by mapping time-varying edge signals \( x_{1}[t] \) onto the topology and should not simply be perceived as a static weighted version of the graph Laplacian matrix \( L_0 \). By combining \( \hat{L}_0[t] \) time-varying edge signals, we result in a time-varying influence of \( x_{1}[t] \) on \( x_{0}[t] \) in the form of time-varying regression matrix \( \hat{L}_0[t] \) that models the change of vertex signal \( x_{0}[t] \).

A diffusion process on a graph is when vertex signals propagate through the adjacency relationships to their locally connected neighbors \([41], [42]\). If we break the matrix multiplication in \((22)\) down, we will see that

\[
x_{i}[t + 1] = \sum_{j=1}^{N} \hat{L}_{ij}x_{j}[t],
\]

where \( \hat{L}_{ij} \) is the \( i^{th} \) row and \( j^{th} \) column of \( \hat{L}_0[t] \), and \( x_{i}[t] \) is the \( i^{th} \) element of \( x_{0}[t] \). Realizing that in \( \hat{L}_0[t] \), \( \hat{L}_{ij} \) is non-zero only when there is an edge, each summation in equation \((23)\) essentially is calculating the vertex signal at \( x_{i} \) minus all the neighboring vertex signals from 1-hop away. When repeating \((23)\) for all \( N_0 \) vertices, each vertex will be aggregated to its 1-hop adjacent neighbor. If we look at \((23)\) in a diffusion aspect, it essentially means that all the vertices will be propagated to their connected neighbors, meaning that the aggregation done in \((23)\) is equivalent to a diffusion.

Following the above logic, the \( p^{th} \) power of \( \hat{L}_0 \) will be a time-varying diffusion in the \( p \)-hop neighborhood. Summing all the terms for \( p = 0 \ldots P \) gives us a combined 1-hop to \( P \)-hop regression:

\[
\hat{H}_0[t] = \sum_{p=0}^{P} \hat{\theta}_p \hat{L}_0^p[t],
\]

where \( \hat{\theta}_p \) is introduced as polynomial coefficients. The diffusive nature of this formulation means that missing data on the vertices can be imputed from neighborhood connections as the diffusion progresses. Moreover, when the weights \( \hat{\theta}_p \) in \((24)\) are defined based on a properly designed simplicial filter, simplicial denoising can be achieved with this formulation; a spectral analysis will be conducted in the next subsection as a guide of how to define the weights. The expression in \((24)\) can be realized by the Chebyshev approximation shown in \((7)\). If we plug \( \hat{L}_0[t] \) into \((7)\) and rearrange the terms, the resulting expression becomes a spatial diffusion represented by \( \hat{L}_0[t] \):

\[
\theta_0 + \sum_{p=1}^{P} \hat{\theta}_p T_p(\hat{L}_0[t]) = \sum_{p=0}^{P} \hat{\theta}_p \hat{L}_0^p[t] = \hat{H}_0[t],
\]

where now \( \hat{\theta}_p \) is the polynomial coefficient instead of \( \theta_p \).

Another approach to forming the time-varying vertex regression in \((22)\) is to use spectral filters by setting the SFT in \((2)\) to \( k = 0 \) to form a simplicial (node) filter \( \Sigma_{0,x} \) to approximate \( L_0[t] \). Let us represent the filter \( \Sigma_{0,x} \) using a function \( h(\Lambda_0[t]) = \Sigma_{0,x} \). Then, the regression \((22)\) in terms of the spectral filtering in \((3)\) is

\[
\hat{U}_0[t]h(\Lambda_0[t])\hat{U}_0^T[t]x_{0}[t] = h(\hat{L}_0[t])x_{0}[t].
\]

Applying the logic of using a shifted Chebyshev polynomial \( T_p(L_0[t]) \) similar to what is shown in \((6)\), we can approximate \((26)\) with the \( p \)-hop time-varying regression in \((24)\):

\[
h(\hat{L}_0) \approx \theta_0 + \sum_{p=1}^{P} \theta_p T_p(\hat{L}_0[t]) = \sum_{p=0}^{P} \hat{\theta}_p \hat{L}_0^p[t] = \hat{H}_0[t].
\]

Equation \((27)\) establishes the relationship between the spectral domain and the spatial domain, illustrating the underlying filter \( \Sigma_{0,x} \) can be applied from two different perspectives.

There is a limitation of using spectral methods in practice. The simplicial filter \( \Sigma_{0,x} \) is defined by the frequency set \( \mathcal{F}_0 \) operating on the frequencies (eigenvalues) of \( L_0[t] \). This means that spectral filtering operation in \((26)\) requires the eigendecomposition at every time instance when the signals \( x_1[t] \) change due to the time-varying formulation in \((22)\). Moreover, the eigendecomposition of the graph Laplacian suffers from high computational cost and numerical instabilities when the graph topology is large \([43]\). The same limitation can be found in TSP for the Hodge-Laplacian \([24]\). Thus, the spatial formulations are preferred in practice while the spectral formulations provide a signal processing analytical perspective.

For simplicity, we assume that the topology is known, but the signals on the vertices and edges are noisy or have missing entries at the current time \( t \). Preferably, the topology does not change over time. An example of synthetic data based on a real-world traffic network of the above data model is shown in Fig. 1 where time-varying signals are generated on both the graph vertices and the edges. Even though the formulas in \((24)\) provide diffusion-based vertex regressions under the influence of edge signals, they consider only the diffusion perspective but not the online estimation perspective. We will improve this formulation by considering the temporal changes to achieve adaptive estimation.

B. Joint estimation of vertex and edge signals

AJVEE is targeted at cases where time-varying edge signals influence time-varying vertex signals. We can specify the ALMS-Hodge onto the graph edges by specifying \( k = 1 \),
resulting in $H_1 = \sum_{p=0}^P \theta_{1,p} L_p^0$. Using Chebyshev approximation, the AJVEE update based on the ALMS-Hodge on the edge signals can then be obtained as:

$$\hat{x}_1[t+1] = \hat{x}_1[t] + \mu_1 \sum_{p=0}^P \hat{\theta}_{1,p} L_p^0 e_1[t] + r_1, s R_{1,S} \hat{x}_1[t].$$  \hspace{1cm} (28)$$

To define the AJVEE expression on the graph vertices, we specify $k = 0$ in the ALMS-Hodge. However, AJVEE will be following our time-varying regression data model defined in Section [IV-A] on the graph vertices. The vertex signal $x_0[t]$ is under the influence of the time-varying edge signal $x_1[t]$ so the regression parameters are the current step edge signal estimate $\hat{x}_1[t]$ obtained at the previous time step $t-1$. As a result, the matrix $H_k$ in ALMS-Hodge in (14) is replaced with the time-varying vertex regression $H_0[t] = \sum_{p=0}^P \theta_{0,p} L_p^0$ to incorporate the influence of the time-varying edge signal $x_1[t]$ onto the time-varying vertex signals $x_0[t]$. The vertex update strategy in AJVEE based on ALMS-Hodge is

$$\hat{x}_0[t+1] = \hat{x}_0[t] + \mu_0 \sum_{p=0}^P \hat{\theta}_{0,p} (L_0^0[t])^p e_0[t] + r_0, s R_{0,S} \hat{x}_0[t],$$  \hspace{1cm} (29)$$

where $\hat{x}_0[t]$ is the estimation of vertex signals at time $t$, and the residual $e_0[t] = y_0[t] - x_0[t]$ is the estimation error.

Instead of separately estimating the vertex and edge signals, a joint estimation can be done by the simultaneous operation of (28) and (29). We define a joint operator $H_J[t]$ to incorporate the adjacent relationship within one simplex order together with the incidence relationship between different orders of simplicity. Assuming that the signals are on the vertices (0-simplex) and edges (1-simplex), and also assuming that we have the triangle (2-simplex) adjacency but not the 2-simplicial signals, we can formulate such joint shift operator on the error terms $e_0[t]$ and $e_1[t]$ by

$$H_J[t] = \text{blkdiag}(\mu_0 \hat{H}_0[t], \mu_1 H_1),$$  \hspace{1cm} (30)$$

where \text{blkdiag}(\cdot) is the block-diagonal concatenation of two matrices. Similarly, the signal aggregation matrices are concatenated as

$$R_{J,S} = \text{blkdiag}(r_0, s R_{0,S}, r_1, s R_{1,S}) \text{ and }$$

$$R_{J,S} = \text{blkdiag}(r_0, s R_{0,S}, r_1, s R_{1,S}).$$  \hspace{1cm} (31)$$

Accordingly, the signals, observations, and the estimation error should also be concatenated:

$$\hat{x}_J[t] = \text{vec}(\hat{x}_0[t], \hat{x}_1[t]),$$

$$y_J[t] = \text{vec}(y_0[t], y_1[t]),$$

$$e_J[t] = \text{vec}(e_0[t], e_1[t]).$$  \hspace{1cm} (32)$$

The update function of the AJVEE can be constructed as

$$\hat{x}_J[t+1] = \hat{x}_J[t] + H_J[T] e_J[t] + R_{J,S} \hat{x}_J[t] + R_{J,S} \hat{x}_J[t].$$  \hspace{1cm} (33)$$

The individual vertex signal and edge signal can be obtained by mapping the corresponding elements in $\hat{x}_J[t+1]$ back into $x_0[t+1]$ and $x_1[t+1]$.

The AJVEE operates in an online fashion where the time-varying edge signal and the time-varying vertex signal are estimated jointly using (33) at each time instance. AJVEE simultaneously performs missing signal imputation as it makes predictions because of the incorporation of simplicial diffusion. Additionally, from the spectral filter aspect, the simplicial filter within AJVEE ensures denoising is done along the update. These advantages and characteristics of AJVEE enable AJVEE to overcome the challenges of representing topological irregularity, estimating unknown data from known data, removing noise present in the data, and extracting cross-space-time variations we pointed out at the beginning of Section [II]. The complete AJVEE algorithm is summarized in Algorithm [1]. For the spectral analysis of the AJVEE from the TSP perspective, please refer to Appendix [B].

**Algorithm 1** Adaptive joint vertex-edge estimation

Given the (unweighted) graph topology $G[t]$, spectral filters $\Sigma_0$ and $\Sigma_1$, Initialize $\hat{x}_0[0]$ and $\hat{x}_1[0]$ Construct $H_1$ and $H_0[t]$ Construct $R_1$ and $R_0$ based on prior knowledge using (16), (17), (19), or (20) Construct $H_J[t]$ and $R_J[t]$ using (31)

while there are new observations $y_J[t]$ and $y_1[t]$

Define the changed components in $\hat{H}_J[t]$ and $R_J$ again

Output joint signal $\hat{x}_J[t+1]$ from $y_J[t]$ using (33)

end while

C. Generalizing AJVEE to high-ordered simplicial complexes

In this section, the AJVEE is defined given a more generalized scenario where the time-varying simplicial signals on $X_k$ are influenced by the time-varying signals simplicial signals on $X_{k+1}$. Given the topology, we can rely on defining the ALMS-Hodge on $X_k$ and $X_{k+1}$ shown in Section [III]. First, the $k$-simplex regression in (22) will need to have the parameters defined using the $k+1$-simplex. Then, we can define the joint terms

$$\hat{x}_J[t] = \text{vec}(\hat{x}_k[t], \hat{x}_{k+1}[t]),$$

$$y_J[t] = \text{vec}(y_k[t], y_{k+1}[t]),$$

and

$$e_J[t] = \text{vec}(e_k[t], e_{k+1}[t]).$$  \hspace{1cm} (34)$$

where $\hat{x}_J[t]$ is the joint time-varying $X_k$ and $X_{k+1}$ signals, $y_J[t]$ is the joint observations, and $e_J[t]$ is the joint estimation error. To formulate the proper AJVEE on simplicial complexes that are higher-ordered compared with graph vertices and edges, we will have matrices defined using the TSP techniques discussed in Section [III]

$$H_J[t] = \text{blkdiag}(\mu_k \hat{H}_k[t], \mu_{k+1} H_{k+1}),$$

$$R_{J,S} = \text{blkdiag}(r_k, s R_{k,S}, r_{k+1}, s R_{k+1,S}) \text{ and }$$

$$R_{J,S} = \text{blkdiag}(r_k, s R_{k,S}, r_{k+1}, s R_{k+1,S}).$$  \hspace{1cm} (35)$$

The update scheme of AJVEE still follows (33) after plugging the above matrices into (33). However, conceptually, because we are defining all operations using the Hodge Laplacian and TSP, for $k \neq 0$, AJVEE is no longer defined on the vertices.
and edges. For example, if \( k = 2 \), then AJVEE becomes an algorithm that jointly estimates the time-varying signals on the triangles (2-simplex) that are under the influence of the time-varying signals on the tetrahedrons (3-simplex). The general case of AJVEE is illustrated in Algorithm 2.

Algorithm 2 Joint estimation of signals in \( X_k \) and \( X_{k+1} \)

Given: (unweighted) graph topology \( G[t] \), spectral filters \( \Sigma_k \) and \( \Sigma_{k+1} \)

Initialize \( \hat{x}_k[0] \) and \( \hat{x}_{k+1}[0] \)

Construct \( H_k \) and \( H_{k+1}[t] \)

Construct \( R_k \) and \( R_{k+1} \) based on prior knowledge using (16), (17), (19), or (20)

Construct \( H_J[t] \) and \( R_J[t] \) using (35)

while there are new observations \( y_k[t] \) and \( y_{k+1}[t] \) do

Define the changed components in \( y_k[t] \) and \( y_{k+1}[t] \)

Output joint signal \( \hat{x}_J[t+1] \) from \( y_J[t] \) using (33)

end while

V. EXPERIMENT RESULTS AND DISCUSSION

A. Experiment settings

The datasets we conduct the experiments on are the Sioux Falls transportation network, the Anaheim transportation network, and the England COVID-19 dataset. All the experiments are conducted in MATLAB 2022a and are repeated \( N = 100 \) times unless specified otherwise. In the experiment, all the tested algorithms will use lowpass filters defined in (8). The diffusion initialization strategy proposed in (30) is used to initialize the signals for all the experiments. All algorithms running on the same task will have the same passband to ensure the fairness of comparison. The performance of all the experiments is measured using the normalized mean squared error (NMSE):

\[
\text{NMSE}_{k[t]} = \frac{1}{N} \sum_{i=1}^{N} \frac{\| (\hat{x}_k[i][t] - x_k[i][t]) \|_2^2}{\| (x_k[i][t]) \|_2^2},
\]

where \( \| (\hat{x}_k[i][t] - x_k[i][t]) \|_2^2 \) is the squared error on the \( i^{th} \) run.

The topology for the Sioux Falls transportation network and the Anaheim transportation network are gathered from the real world (44). The Sioux Falls network consists of \( N_0 = 24 \) vertices and \( N_1 = 38 \) edges and the Anaheim network has \( N_0 = 406 \) vertices and \( N_1 = 624 \) edges. These two transportation networks have provided static traffic flows on the edges for the traffic under equilibrium (44). The vertex and edge signal missing rates for the Sioux Falls network is 26% and for the Anaheim network is 30%. The missing signal observations in the Sioux Falls network are modeled using the spectral sampling strategy discussed in (6) and the missing signal observations in the Anaheim network are set to be random for each experiment run but will remain fixed within an experiment run. The edge signals are generated synthetically by combining several sinusoidal functions to make the static signal provided by the dataset time-varying:

\[
x_1[t] = \sum_{p=1}^{P} \alpha_p f_p[t],
\]

where the signals \( \alpha_p \) are static edge signals and the term \( f_p[t] \) is a sinusoidal function. The ground truth vertex signals are synthetically generated with additive i.i.d. Gaussian noise \( \epsilon_0[t] \) with zero mean and standard deviation of \( \sigma = 0.2 \) on both the Sioux Falls network and the Anaheim network:

\[
x_0[t+1] = \hat{H}_0[t] x_0[t] + \epsilon_0[t].
\]

We should point out that even though the vertex signal is synthetic, which seems a bit abstract, it is built using properties extracted from the underlying real-world topology. In the context of a transportation network, one may perceive the vertex signal as a measurement of congestion at an intersection or a number of the crowd at a place of interest. The magnitude of the vertex signal will be proportional to the amount of traffic on the edges. Using AJVEE, we aim to predict the congestion or population on the vertices using the predicted traffic on the edges. Since this paper is not focused on transportation network models, we will not discuss the meaning of vertex signals further and will use them purely as the target signal to be estimated. Figure 1 illustrates an example of such a time-varying vertex signal under the influence of time-varying edge signals on the Sioux Falls network.

The England COVID-19 dataset consists of a series of 61 dynamic graphs (36). In the experiment, we perceive every newly appeared edge at the topology in time \( t \) compared to the previous topology at \( t - 1 \) as an edge with signals missing, meaning that the corresponding diagonal entry in \( D_{1,t} \) is 0. All 61 graphs have \( N_0 = 129 \) vertices, with a different number of edges \( N_1[t] \) on each day. The vertices represent different regions in England, recording the number of COVID-19 cases in the region each day, and the edges represent population mobility between regions; self-loops represent the mobility within the region. The upper row of Figure 7 displays the England COVID-19 dataset at three different time points after our preprocessing. Even though there is no data on the 2-simplex in any of the datasets, the upper adjacency of the edges can still be represented in \( L_{1,u} \) because the topological structures still exist.

The AJVEE proposed in Algorithm 2 will be examined on the ability to jointly reconstruct time-varying vertex and edge signals under the conditions that the edge signal influences the vertex signal. Since ALMS-Hodge is the building block of AJVEE, we will conduct preliminary experiments on the ALMS-Hodge for both steady-state and time-varying simplicial signals in the \( k \)-simplices prior to the AJVEE experiments.

B. ALMS-Hodge analysis

Since the ALMS-Hodge is a component within AJVEE, we first conduct a few experiments to assess the performance of our ALMS-Hodge procedure in (14) at estimating a time-varying edge signal in an online procedure on the Sioux Falls network. To specify (14) onto the edges, we will be setting \( k = 1 \), resulting in the edge signal estimation formula shown in (28). The ALMS-Hodge is tested under the following two experiment settings. First, fix the noise level and run the ALMS-Hodge for 4 different step sizes. Second, fix the step size and run the ALMS-Hodge for 4 different noise levels.
missing edge signal observations is included in Figure 2a and denoted as noisy & missing. From the edge signal estimation in Figure 2a, we can see that the step size parameter $\mu_1$ indeed controls the magnitude of the update. For $\mu_1 = 0.1$, the magnitude of the update is too small, which is reflected by visually inspecting Figure 2a. The estimation does not catch up with the time-varying change in the signal. As for the case when $\mu_1$ is too large ($\mu_1 = 1.8$), the large magnitude causes the estimation to fluctuate and unstable, which can be observed from Figure 2a as well. This confirms that $\mu_1$ controls the amount of update of the ALMS-Hodge. The error of the estimation will be large when $\mu_1$ is either too large or too small. Inspecting the NMSE in Figure 2b, we can gain some insights on the design choice of $\mu_1$.

Now, we will be fixing the step size $\mu_1 = 0.6$ and conducting an online estimation of a time-varying edge signal under four different noise settings. The Gaussian noise all have zero mean but four different variances $\sigma = 0.1, 0.3, 0.5$, and $0.7$. The signal reconstruction of one missing edge is shown in Figure 3a and the denoising of one observed edge signal is shown in Figure 3b. Upon examination of Figure 3, we verified that the ALMS-Hodge can effectively conduct online estimations of time-varying signals on the graph edges under Gaussian noise with various noise levels.

### C. Joint Estimation on transportation network

Returning to the question of joint estimation, we will be testing AJVEE based on Algorithm 1 at jointly estimating time-varying vertex and edge signals by utilizing AJVEE within the ALMS-Hodge on the Sioux Falls network and the Anaheim network.

To show the advantages of using a TSP approach (the ALMS-Hodge) in AJVEE, we created a comparison algorithm that uses Line Graph LMS [45] instead of using the ALMS-Hodge in the AJVEE. We denote this approach as LGLMS in later experiments. Different from the TSP-based ALMS-Hodge, the LGLMS uses line graphs to map edge signals on the vertices then processed by GSP. The aggregations terms used by AJVEE in this experiment are (16) for edge signals and (17) for vertex signals. Since the signal aggregation terms are missing in the LGLMS, we added the same signal aggregation terms that we use in AJVEE to LGLMS. For edge signal estimation baselines, in addition to the LGLMS algorithm, we also included a non-adaptive TSP baseline using a basic low pass simplicial filter $\Sigma_{1, x_p}$ similar to (3): $\hat{x}_1[t + 1] = U_1 \Sigma_{1, x_p} U_1^T \hat{x}_1[t]$. The last edge signal baseline is a Moving Average (MA) of the past 5 observations.

The vertex estimation results of AJVEE will be compared with the following three GSP baselines: the GLMS algorithm [41], the adaptive Graph Least Mean $p^{th}$ order algorithm (GLMP) [26], and the adaptive Graph-Sign (G-Sign) algorithm [29]. Note that by the algorithm definitions in their original literature, the GSP methods are using a fixed graph Laplacian, so among all the baselines only the LGLMS algorithm is aware that the time-varying edge signals influence the vertices. In addition, we will feed the edge signal estimation of LGLMS into the vertex estimation scheme of the ALMS-Hodge as a
comparison to purely using the ALMS-Hodge in AJVEE. The MA is also included for the vertex signal estimation.

The step sizes $\mu_k$ and the aggregation weights $r_k$ are tuned using grid search. As for the filter used in the experiment, we again use an ideal low-pass filter defined by (8) on both the vertices and the edges. The final parameter selections for AJVEE are shown in Table II. To promote fair comparison, all the tested algorithms share the same $\mu_0$ for vertex signal estimation and all the tested algorithms share the same $\mu_1$ for edge signal estimation. Similar principle applies to the AJVEE and LGLMS for the aggregation weights. The NMSE of the signal estimation results over the 100 runs are recorded in Fig. 4 for the Sioux Falls network and in Fig. 5 for the Anaheim network. Additionally, an illustration of the estimation results on one of the unobserved edges and one of the unobserved vertices in the Sioux Falls network is shown in Fig. 4.

![Fig. 4: Estimation on one of the unobserved edges (top) and one of the unobserved vertices (bottom) of the Sioux Falls network.](image)

From Fig. 4 and Fig. 5, AJVEE has lower NMSE than the baselines for the majority of the time points for the edge estimation task. From the low NMSE, we can confirm that the AJVEE is accurate at tracking the changes of time-varying edge signals. AJVEE outperforms the LGLMS algorithm because the LGLMS does not operate directly on the edge signal; the LGLMS algorithm treats the original edge signal as vertex signals on a line graph projection. However, the projected signal in LGLMS is not guaranteed to be smooth on the 0-simplices, so using the GSP nodal operations to predict edge signals based on a smooth graph assumption and a low-pass filter on the line graph projection does not perform well [46]. As for the TSP non-adaptive method, the filtering operation $U_L^T \Sigma_1 x_p U_L^T y_L[t]$ is not optimized to reduce time-dependent errors compared to the adaptive update $H_1 e_1[t]$, which lead to the less compelling results compared to AJVEE. The MA has a relatively high NMSE because it can not make predictions on the missing edges, which can be observed from Figure 4 that the MA consistently outputs 0 for unobserved signals. The other graph algorithms can infer the missing edge signals because each of the updates implies spatial diffusion, which means that these algorithms conduct spatial missing signal imputation and temporal 1-step predictions at the same time.

The AJVEE has the lowest estimation error for the vertex signal estimation shown in Fig. 5 and Fig. 6. The GSP approaches do not quite match the performance of AJVEE in predicting the time-varying vertex signals because the
GSP methods used a fixed graph Laplacian to process the signals on the vertices. Here, notice that the LGLMS performs only slightly worse than AJVEE. The reason behind this is because, first, here in the vertex estimation the LGLMS is actually feeding the edge estimation results of LGLMS into the vertex update portion of the AJVEE, so here the LGLMS is essentially another variant of AJVEE. Second, the LGLMS estimation also benefited from the aggregation terms derived for ALMS-Hodge. On the other hand, AJVEE utilizes the time-varying edge signal to represent the diffusion dynamics of the vertex signals, which is more accurate at tracking the temporal evolving patterns of the time-varying vertex signals. Our proposed AJVEE framework in Algorithm\textsuperscript{1} has demonstrated the ability to effectively capture the time-varying edge signals jointly with the time-varying vertex signals. LGLMS takes a similar approach as AJVEE to use edge signals to represent the time-varying regression, but the relatively inaccurate edge signal estimation causes the LGLMS to have inaccurate vertex signal estimation.

D. COVID-19 case forecasting in population mobility network

In this last experiment, we will use AJVEE to jointly estimate real-world time-varying vertex and edge signals in the England COVID-19 dataset using the procedure in Algorithm\textsuperscript{1}. The goal of this real-world dataset is to predict the confirmed number of COVID-19 cases in the future using the procedure in Algorithm\textsuperscript{1}. The England COVID-19 dataset is modeled by a dynamic sampling matrix in the edges $D_{1,S}[t]$ where we compare the graph topology of the graph at the current time point $G[t]$ with newly obtained graph topology $G[t+1]$. If an edge is present in both $G[t]$ and $G[t+1]$, then the corresponding diagonal element in $D_{1,S}[t+1]$ will be 1. If an edge is present in $G[t+1]$ but not in $G[t]$, then the corresponding diagonal element in $D_{1,S}[t+1]$ will be 0. Both aggregations (16) and (17) are adopted in the experiment for predicting population mobility. The predicted number of confirmed cases by AJVEE will be compared against three online GSP algorithms: GLMS \cite{34}, GLMP \cite{35}, and G-Sign \cite{29}. All these GSP algorithms are modified to be aware of the topology change, but they are not provided with the time-varying population mobility on the graph edges. The choice of this baseline is to demonstrate the usefulness of acquiring the population mobility at predicting the number of cases.

We will also be using two simple baselines: 1. the MA of confirmed cases from the past 5 days; 2. repeat the noisy case count from the previous day (denoted as Last-day). This experiment reflects a real-world example of utilizing the time-varying signals on one simplicial dimension as an influence factor upon another simplicial dimension.

The predicted population mobility and the predicted number of cases at three different time points using Algorithm\textsuperscript{1} are illustrated in Fig.\textsuperscript{7}. From Fig.\textsuperscript{7} it is clear that the recovered population mobility and the number of cases are close to the ground truth. In Fig.\textsuperscript{8} we calculated the average absolute error of the predicted number of cases at each time point in all the vertices, which is the suggested metric by the dataset\textsuperscript{16}. By inspecting Fig.\textsuperscript{8} we observe that our joint estimation framework has the lowest error most of the time. AJVEE outperforms GSP algorithms that purely use vertex information because the GSP algorithms are unable to modify the magnitude of the update based on the information from the population mobility. However, because AJVEE jointly estimates the edge and vertex signals, AJVEE is more aware of the change in the vertices that are caused by the change in the edges. The nonstationary behavior of the confirmed cases causes relatively worse performance of MA and Last-day. The results in this experiment indicate that our proposed AJVEE framework could utilize and jointly predict the dynamic information on the edges and the vertices to solve a real-world problem of online prediction of time-varying signals. The current approach to this real-world problem is a bit primitive. In the next section, we will discuss potential improvements to our AJVEE framework to solve similar problems.

VI. CONCLUSION, LIMITATION, AND FUTURE WORK

This paper presented the AJVEE framework for estimating time-varying vertex signals and edge signals, in which the dynamics of the vertex signals are in the form of time-varying edge signals. The underlying online estimation method of AJVEE is the ALMS-Hodge, an adaptive procedure backed by spatial and spectral TSP techniques. By treating graphs as a special case of a simplicial complex, two ALMS-Hodge
with different simplicial orders are specified on the vertices and edges and then merged into the joint estimation. Even though AJVEE is proposed on the vertices and edges, we also discussed a more generalized setting where AJVEE is extended to higher-order structures. In summary, AJVEE effectively addresses the challenges of representing topological irregularity, estimating unknown data from known data, removing noise present in the data, and extracting cross-space-time variations. Experiments on both synthetic and real data indicate that AJVEE can efficiently solve the task of joint online estimation of time-varying vertex and edge signals.

Currently, one limitation of AJVEE is that the influence of edge signals is applied globally to all vertex signals using operations defined by the Hodge-Laplacian, without considering the direction. Even though the sign and directions at the time of edge estimation, when casting the edge signals as the influence of the vertex signals, the direction and sign information are not properly utilized, leading to an invariant model. However, these directional information and sign differences in edge signals, which can change over time, can be important to the data. The current invariant treatment of edge signals can lead to sub-optimal performance in downstream tasks that depend on these directional and sign attributes. We recognize this as a challenge and suggest that future work should explore equivariant models to better account for these variations, rather than relying on the current invariant model. Another limitation in our current setup is that we are not able to process directed multi-edges between nodes. Additionally, the underlying topology is assumed to be known. One possible direction worth exploring is combining our AJVEE framework with topology learning algorithms. This combination will further extend the predictive power of AJVEE onto graphs that dynamically evolve the topology over time. The extension of TSP approaches to dynamic graphs, multi-graphs, and equivariant models will be interesting future directions. Due to the lack of suitable real-world time-varying traffic flows for our experimental setting, we utilized a combination of synthetic data generated from real data on top of a graph representing real-world road networks. This is a limitation in the current experiment design and we look forward to applying our AJVEE framework to additional real-world datasets that contain real time-varying features.

REFERENCES

[1] A. Sandryhaila and J. M.F. Moura, “Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure,” IEEE Signal Process. Mag., vol. 31, no. 5, pp. 80 – 90, 2014.
[2] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” IEEE Signal Process. Mag., vol. 30, pp. 83 – 98, 2013.
[3] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and Pierre Vandergheynst, “Graph signal processing: Overview, challenges, and applications.” Proc. IEEE, vol. 106, no. 5, pp. 808-828, 2018.
[4] P. D. Lorenzo, S. Barbarossa, P. Banelli, and S. Sardellitti, “Adaptive least mean squares estimation of graph signals,” IEEE Trans. Signal Inf. Process. Netw., vol. 2, no. 4, pp. 555 – 568, 2016.
[5] C. Wang, Y. Zhu, T. Zhang, H. Liu, and J. Yu, “Modeling inter-station relationships with attentive temporal graph convolutional network for air quality prediction,” in WSDM, 2021, p. 616–634.
[6] M. J. M. Spelta and W. A. Martins, “Normalized lms algorithm and data-selective strategies for adaptive graph signal estimation,” Signal Process., vol. 167, pp. 107326, 2020.
[7] S. Ancherbak, E. E. Kuruoglu, and M. Vingron, “Time-dependent gene network modelling by sequential monte carlo,” *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 13, no. 6, pp. 1183–1193, 2016.

[8] W. Huang, T. A. W. Bolton, J. D. Medaglia, D. S. Bassett, A. Ribeiro, and D. Van De Ville, “A graph signal processing perspective on functional brain imaging,” *Proceedings of the IEEE*, vol. 106, no. 5, pp. 868–885, 2018.

[9] M. M. Li, K. Huang, and M. Zitnik, “Graph representation learning in biomedicine and healthcare,” *Nat. Biomed. Eng.*, pp. 1–17, 2022.

[10] J. Yang and J. Peng, “Estimating time-varying graphical models,” *J. Comput. Graph. Stat.*, vol. 29, no. 1, pp. 191–202, 2020.

[11] J. Jia, M. T. Schaub, S. Segarra, and A. R. Benson, “Graph-based semi-supervised and active learning for edge flows,” in *KDD*, 2021, pp. 761–771, ACM.

[12] K. Han, G. Eve, and T. L. Friesz, “Computing dynamic user equilibria on large-scale networks with software implementation,” *Networks and Spatial Economics*, vol. 19, no. 3, pp. 869–902, 2019.

[13] Q. Yang, Y. Lyu, X. Li, C. Chen, and F. L. Lewis, “Adaptive distributed synchronization of heterogeneous multi-agent systems over directed graphs with time-varying edge weights,” *J. Franklin Inst.*, vol. 358, no. 4, pp. 2434–2452, 2021.

[14] M. Yang, E. Isufi, and G. Leus, “Simplicial convolutional neural networks,” in *ICASSP*, 2022, pp. 8847–8851.

[15] J. Hong, Y. Yan, E. E. Kuruoglu, and W. K. Chan, “Multivariate time series forecasting with GARCH models on graphs,” *IEEE Trans. Signal Process.*, vol. 9, pp. 557–568, 2023.

[16] S. Barbarossa and S. Sardellitti, “Topological signal processing over simplicial complexes,” *IEEE Trans. Signal Process.*, vol. 68, pp. 2992–3007, 2020.

[17] S. Barbarossa and S. Sardellitti, “Topological signal processing: Making sense of data building on multiway relations,” *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 174–183, 2020.

[18] T. M. Roddenberry, F. Frantzen, M. T. Schaub, and S. Segarra, “Topological signal processing with the dirac operator,” in *Asilomar*, 2022.

[19] S. K. Kadambari, R. Francis, and S. P. Chepuri, “Distributed denoising for graph signal processing,” in *DCCOS*, 2011, pp. 1–8.

[20] B. Stabler, H. Bar-Gera, and E. Sall, “Transportation networks for research,” url = https://github.com/bstablerto/TransportationNetworks.

[21] Y. Yan and E. E. Kuruoglu, “Online signal estimation on the graph edges via line graph transformation,” *preprint*, arXiv:2311.00656, 2023.

[22] F. Krishnan, R. Money, B. Beferull-Lozano, and E. Isufi, “Simplicial vector autoregressive model for streaming edge flows,” in *ICASSP*, 2023.

[23] M. T. Schaub, Austin R. Benson, P. Horn, G. Lippner, and A. Jadbabaie, “Random walks on simplicial complexes and the normalized hodge 1-laplacian,” *SIAM Review*, vol. 62, no. 2, pp. 353–391, 2020.

[24] C. Battiloro, P. Di Lorenzo, and S. Barbarossa, “Topological slepians: Maximally localized representations of signals over simplicial complexes,” in *ICASSP*, 2022, pp. 5932–5936.

[25] D. I. Shuman, P. Vandergheynst, and P. Frossard, “Chebyshev polynomial approximation for distributed signal processing,” in *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, 2022.

[26] Y. Yan, H. C. So, and E. E. Kuruoglu, “Vertex-frequency analysis of graph signals in the edge-space,” in *GlobalSIP*, 2018, pp. 735–739.

[27] L. Stanković and E. Sejdić, *Vertex-frequency analysis of graph signals*, Springer, 2019.

### APPENDIX A

#### ALMS-HODGE CONVERGENCE ANALYSIS

To make sure the ALMS-Hodge procedure in the AJVEE outputs a stable estimation, we analyze the update of the ALMS-Hodge under steady-state simplicial signal estimation. For ease of analysis, we will not be using the polynomial approximation of $H_k$ shown in (27) in this analysis but rather use the spectral definition $H_k = (\tilde{D}_k \tilde{F}_k)^{-1} \tilde{U}_k \tilde{F}_k$. Suppose that we have a clean simplicial signal $\tilde{x}_k$, the update error $\tilde{x}_k[t]$ at time $t$ is the difference between the current estimation $\tilde{x}_k[t]$ and $\tilde{x}_k$: $\tilde{x}_k[t] = \tilde{x}_k - \tilde{x}_k[t]$. The ALMS-Hodge is stable if the update error converges under steady state estimation. To begin the proof, we use the SFT to transfer the update function (14) in the spectral domain:

$$\tilde{s}_k[t+1] + \mu_k \tilde{U}_k F_k H_k e_k[t] + r_k \tilde{U}_k F_k R_k \tilde{U}_k F_k \tilde{s}_k[t],$$  \hspace{1cm} (39)

Since the aggregation term is recursive based on $\tilde{s}_k[t]$, we will first prove the update error $\tilde{s}_k[t]$ converges independent of the aggregation term. The estimation error $e_k[t]$ can be written in terms of $\tilde{s}_k[t]$ and $\tilde{s}_k$ in the spectral domain as $e_k[t] = D_{\tilde{S}} \tilde{s}_k[t] + \eta_k[t] - \tilde{s}_k[t]$. Then, dropping the aggregation, (39) is rearranged into

$$\tilde{s}_k[t+1] = \Omega \tilde{s}_k[t] + \mu_k \tilde{U}_k F_k H_k D_k S \eta_k[t],$$  \hspace{1cm} (40)

where $\Omega = (1 - \mu_k \tilde{U}_k F_k H_k D_k S \tilde{F}_k)$. Equation (40) can be expressed as a recursive update starting from $t = 0$:

$$\tilde{s}_k[t+1] = \Omega^t \tilde{s}_k[0] + \mu_k \sum_{i=1}^{t} \Omega^{i-1} \tilde{U}_k F_k H_k D_k S \eta_k[i].$$  \hspace{1cm} (41)
Now, we can find the update error in the mean squared sense by calculating the variance of (41):
\[
E\|\tilde{s}_k[t+1]\|^2 = E\|\tilde{s}_k[0]\|^2 + \mu_k^2 E \sum_{i}^1 (U^T_{k,r}H_kD_k,S\eta_k[t])^T \Phi^i (U^T_{k,r}H_kD_k,S\eta_k[t]).
\] (42)

For notation simplicity, we set \( \Phi = \Omega^T \Omega \) and use the weighted Euclidean norm \( \| \tilde{s}_k[0] \|^2_{\Phi_i} = \| \tilde{s}_k[0] \|^2_{\Phi^i} \) in (42). Then, we rearrange some terms by using the Trace trick \( E(X^TYY) = Tr(E(XX^T Y)) \), leading to
\[
E\|\tilde{s}_k[t+1]\|^2 = E\|\tilde{s}_k[0]\|^2 + \mu_k^2 \sum_{i}^1 Tr(\Pi^T \Phi_i),
\] (43)
where \( \Psi = U^T_{k,r}H_kD_k,S C_k H_k^T U_{k,r} \) and \( C = \text{cov}(\eta, \eta) \). With some further derivations using two properties \( \text{Tr}(XY) = \text{vec}(X^T) \text{vec}(Y) \) and \( \text{vec}(XYZ) = (Z^T \otimes X) \text{vec}(Y) \), we can factor out the terms independent of the summation index \( i \) from (43). The summation in (43) becomes a geometric series when the condition \( \| \Omega \| < 1 \) is satisfied, then
\[
E\|\tilde{s}_k[t+1]\|^2 = E\|\tilde{s}_k[0]\|^2 + \mu_k^2 \sum_{i}^1 \text{vec}(\Psi)^T \Pi \text{vec}(I),
\] (44)
where \( Q = \Omega^T \otimes \Omega \). Taking the limit with \( \lim_{t \to \infty} E\|\tilde{s}_k[t+1]\|^2 \) and combining the condition of \( \| \Omega \| < 1 \), we recognize the right side of (44) converges to a constant. The only user-defined parameter in the ALMS-Hodge without aggregation is a positive step size \( \mu_k \). Using the properties of \( l_2 \)-norm, we can have an inequality \( \| \Omega \| < \| I \| + \mu_k \| (U^T_{k,r}H_kD_k,S U_{k,r}) \| < 1 \). Now, because the sampling matrix \( D_k,S \) is idempotent and self-adjoint, the matrix \( U^T_{k,r}H_kD_k,S U_{k,r} \) is symmetric, and its \( l_2 \)-norm will be less or equal to its largest eigenvalue. This means that the ALMS-Hodge has a bounded update error and the algorithm converges if \( \mu_k \) is chosen under the condition
\[
0 < \mu_k < 2 / \lambda_{\max}(U^T_{k,r}H_kD_k,S U_{k,r}),
\] (45)
where \( \lambda_{\max}(\cdot) \) calculates the maximum eigenvalue of a matrix. Now, with the selection condition of \( \mu_k \) provided, let us look back into (44). If we choose a smaller value for \( \mu_k \), the term \( \Omega \) will be large, so the squared error will converge to a smaller value but will take more iterations to converge. On the other hand, if we choose a larger value for \( \mu_k \), \( \Omega \) is small, and the algorithm converges faster but with higher error.

The ALMS-Hodge converges with the aggregation term. To see this, an expression similar to (40) but with the aggregation can be formulated as
\[
\tilde{s}_k[t+1] = \Omega_r \tilde{s}_k[t] + \mu_k U^T_{k,r}H_kD_k,S \eta_k[t] + r U^T_{k,r}R_k U_{k,r} \tilde{s}_k,
\] (46)
where \( \tilde{s}_k = U^T_{k,r}x_k \) and \( \Omega_r = (\Omega + r U^T_{k,r}R_k U_{k,r}). \) Let \( \Phi_r = \Omega_r^T \Omega_r \) and \( Q_r = \Omega_r^T \otimes \Omega_r \), then take the same derivation steps as before, the update error in the mean squared sense is
\[
E\|\tilde{s}_k[t+1]\|^2 = E\|\tilde{s}_k[0]\|^2 + \mu_k^2 \text{vec}(\Psi_r)^T \Pi \text{vec}(I),
\] (47)
where \( \Psi_r = \Omega_r^T \Omega_r \) and \( Q_r = \Omega_r^T \otimes \Omega_r \). The condition for (47) to converge is again \( \| \Omega_r \| < 1 \). Now, using the property of the \( l_2 \)-norm, we have the inequality \( \| \Omega_r \| < \| I \| + \| r U^T_{k,r}R_k U_{k,r} \| \). To be on the safe side, and assuming that \( r > 0 \), the ALMS-Hodge converges if (45) is met and if we have \( \| \Omega \| + \| r U^T_{k,r}R_k U_{k,r} \| < 1 \). As a result, the range of choice of selecting parameter \( r \) is
\[
0 < r < \frac{1 - \Omega}{\lambda_{\max}(U^T_{k,r}R_k U_{k,r})}.
\] (48)

In order to observe the convergence behavior of the ALMS-Hodge, we will make online estimations for 1200 iterations where at each iteration we have a new noisy and missing observation of the steady-state edge signal. Gaussian distribution with mean \( 0 \) and variance \( 0.18 \) will be used for the noise distribution \( \eta[t] \). The step size is set to \( \mu_1 = 0.1 \). The low pass filter used in the experiment has the pass band of 26 lowest frequencies. Both spatial ALMS-Hodge with order P (denoted as Cheb P) and spectral ALMS-Hodge will be tested. We will be testing the ALMS-Hodge for 6 different settings: spectral ALMS-Hodge, Cheb 3, Cheb 5, Cheb 7, Cheb 9, and Cheb 11. The NMSE of this steady-state experiment is shown in Fig. [9].

![Fig. 9: Steady-state estimation on the edges of the Sioux Falls network using ALMS-Hodge with different step sizes.](image-url)

By inspecting Fig [9] we can observe that all variants of the ALMS-Hodge result in similar NMSE results after 200 iterations. Additionally, all the spatial update functions have similar NMSE performance to the spectral update function when converged.

**APPENDIX B**

**SPECTRAL ANALYSIS OF AJVEE**

There are two incidence matrices we are particularly interested in for the estimation of node and edge signals, namely the node-to-edge incidence matrix is \( B_1 \) and the edge-to-triangle incidence matrix is \( B_2 \). The matrix \( B_1 \) is a divergence operator because \( B_1 x_1[t] \in \mathbb{R}^N \) calculates the difference between the outgoing edge signals and the incoming edge signals at a vertex. An edge signal \( x_1[t] \) is considered divergence-free if \( B_1 x_1[t] = 0 \) [11, 20]. The gradient operator \( B_1^T \)
on the vertex signal $x_0$ calculates the difference between the endpoints in each edge along the edge orientation, inducing a gradient flow $x_G[t] = B_1^T x_0[t] \in \mathbb{R}^N_1$ on the edges. The matrix $B_1^T$ is a curl operator since $B_2 x_1[t] \in \mathbb{R}^N_1$ computes the net flow of the edges along each triangle. If $B_1^T x_1 = 0$ holds true, then the edge signal $x_1[t]$ is curl-free \[22\]. The curl adjoint operator $B_2$ induces the curl flow onto the edges from the triangles by $x_C[t] = B_2 x_2[t]$. From the Hodge decomposition, aggregations that utilize the four abovementioned operators can be defined using (16), (17), (19), and (20).

Taking an edge signal $x_1[t]$ as an example, in the spectral domain, the Hodge decomposition in (18) implies that we can modify the signal $x_1[t]$ and the update error $e_1[t]$ by modifying the harmonic component, the gradient component, and the curl component separately. Based on the Hodge decomposition, the eigenvector matrix $U_1$ and the corresponding eigenvalue matrix $\Lambda_1$ of $L_1$ can be rearranged as shown below \[22\]:

$$U_1 = [U_H, U_G, U_C] \quad \text{and} \quad \Lambda_1 = \text{blkdiag}(\Lambda_H, \Lambda_G, \Lambda_C),$$

(49)

where blkdiag() is the block diagonal operation. The matrices $U_H$, $U_G$, and $U_C$ in (49) are the eigenvector matrix of the harmonic component, the gradient component, and the curl component respectively. The diagonal matrices $\Lambda_H$, $\Lambda_G$, and $\Lambda_C$ contains the eigenvalue corresponds to the eigenvectors in $U_H$, $U_G$, and $U_C$. Because $L_1$ can be represented as the sum of $L_{1,l}$ and $L_{1,u}$, the following expressions can be derived using the logic in (49) \[22\]:

$$L_{1,l} = U_1 \text{blkdiag}(0, \Lambda_G, 0) U_1^T,$$

$$L_{1,u} = U_1 \text{blkdiag}(0, 0, \Lambda_C) U_1^T.$$

(50)

Now, by looking at (49) and (50), it is apparent that by carefully designing the filter $\Sigma_1$ in (15) in the spectral domain, we could achieve the direct modification of the harmonic component, the gradient component, and the curl component of the update $e_1[t]$ as well as the estimated signal $\hat{x}_1[t]$.

A similar analysis can be conducted on the graph vertices by decomposing $L_0$ or $\hat{L}_0[t]$. It should be pointed out that the vertices only have upper adjacency, which simplifies the spectral analysis. Generally speaking, in vertex signal estimation tasks the aggregation requires a smooth (low frequency) signal assumption \[3\]. In the GSP perspective, if $\hat{H}_0[t]$ is defined based on an approximation of a low-pass filter $\Sigma_{F,0}$, then it enforces the estimated signal to be smooth \[47\].