A Time-Vertex Signal Processing Framework

Scalable processing and meaningful representations for time-series on graphs

Francesco Grassi[1], Andreas Loukas[2], Nathanaël Perraudin[2], and Benjamin Ricaud[2]
[1]Politecnico di Torino, Torino, Italy
[2]Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

***Abstract*—An emerging way to deal with high-dimensional non-euclidean data is to assume that the underlying structure can be captured by a graph. Recently, ideas have begun to emerge related to the analysis of time-varying graph signals. This work aims to elevate the notion of joint harmonic analysis to a full-fledged framework denoted as Time-Vertex Signal Processing, that links together the time-domain signal processing techniques with the new tools of graph signal processing. This entails three main contributions: (a) We provide a formal motivation for harmonic time-vertex analysis as an analysis tool for the state evolution of simple Partial Differential Equations on graphs. (b) We improve the accuracy of joint filtering operators by up-to two orders of magnitude. (c) Using our joint filters, we construct time-vertex dictionaries analyzing the different scales and the local time-frequency content of a signal. The utility of our tools is illustrated in numerous applications and datasets, such as dynamic mesh denoising and classification, still-video inpainting, and source localization in seismic events. Our results suggest that joint analysis of time-vertex signals can bring benefits to regression and learning.**

***Index Terms*—Time-Vertex Signal Processing, Graph Signal Processing, Partial Differential Equations**

I. Introduction

Whether examining consensus and rumor spreading over social networks [1]–[3], transportation networks [4] and related epidemic spreading [5], neuronal activation patterns [6] or time-evolving functional network in the brain [7], as well as other datasets collected from a variety of fields, such as physics, engineering, and life-science, much of the high-dimensional data exhibit complex non-euclidean properties.

An emerging way to deal with these issues is to use a graph to capture the structure underlying the data. This has been the driving force behind recent efforts in the signal processing field to extend harmonic analysis to graph signals, i.e., signals supported on the vertices of irregular graphs [8], [9]. In the field of graph signal processing (GSP), the introduction of the graph Fourier transform (GFT) has enabled us to perform harmonic analysis taking into account the structure of the data, and has lead to improvements for tasks such as clustering [10], low-rank extraction [11], spectral estimation [12], [13], non-stationary analysis [14], [15] and semi-supervised learning [16], [17].

Nevertheless, though state-of-the-art graph-based methods have been successful for many tasks, so far they predominantly ignore the time-dimension of data, for example by treating successive signals independently or performing a global average [6], [12], [18]. On the contrary, many of the systems to which GSP is applied to are inherently dynamic. Consider for instance a road network, and suppose that we want to infer traffic conditions given flow information over a subset of highways and streets. Approaches that do not take into account the temporal evolution of traffic will be biased by seasonal variations and unable to provide insights about transient phenomena, such as rush hour traffic, bottlenecks caused by blockage, and stop waves.

Recently, several ideas begin to emerge related to the analysis of time-varying graph signals, such as Joint time-vertex Fourier transform (JFT) [19] and the joint time-vertex filters [20] and filterbanks [21]. While these constitute notable contributions, we argue that the potential of joint harmonic analysis is yet unexplored, both in terms of its foundations, algorithms, and applications.

In this work we aim at elevating the notion of joint harmonic analysis to a full-fledged framework, referred to as the *Time-Vertex Signal Processing Framework*, that links together the time-domain signal processing techniques with the new tools of GSP.

This entails the following contributions:

1. *Connection to PDEs.* We illustrate how joint analysis emerges when analyzing the state evolution of simple Partial Differential Equations (PDEs) on graphs (Section III-A). We also provide an example (epidemic spreading) demonstrating that the joint frequency analysis can be meaningful for the study non-linear and stochastic processes leading to a compact and intuitive representation (Section III-B).
2. *Accurate fast joint filtering.* In Section IV we illustrate the utility of joint filtering time-vertex signals and propose a fast filtering implementation, called Fast Fourier-Chebyshev (FFC) algorithm, which significantly improves upon the state-of-the-art filters both in terms of

separable and non-separable filtering objectives. For the latter case especially, our numerical experiments show that FFC can yield up to two orders of magnitude smaller approximation error at a similar complexity to previous joint filters.

3. *Overcomplete representations.* We study redundant time-vertex dictionaries and exploit them for signal analysis and synthesis. The proposed framework includes a frame condition guaranteeing that no information is lost. Two particular cases are: *time-vertex wavelets* capturing the different scales of the signal components, and the *short time-vertex Fourier transform* that is useful in determining the local time frequency content of the signal.

4. Illustrating the utility of time-vertex analysis. Finally, Section VI provides experimental evidence for the utility of joint harmonic analysis in a number of graph-temporal datasets that were up to now not fully exploited, such as dynamic meshes, video and general dynamics over networks. The range of applications covers the classical signal processing problems of denoising, inpainting and compression, but also extends to feature extraction for classification and source localization problems.

A. Related Work

The time-vertex framework is intimately linked with the stochastic analysis of multivariate signals and, therefore, with graphical models (e.g. [22] and references therein). The main difference between graphical models and GSP lies in the assumption about the relation between the signal and graph [23]. Graphical models adopt a purely Bayesian setting, where edges denote conditional dependencies between variables. As such, the graph usually is a proxy for the covariance and is learned from the data. On the other hand, GSP assumes that the graph is given and its relation to the signal can be understood through harmonic analysis.

In this context, the idea of time-vertex analysis can be traced back to the study [24] aiming to process multi-modal signals with different graphs associated with each of their modalities (i.e., one can consider a time-vertex signal as multi-modal, with time and graph being the two modalities). Collaboration between the graph theory and signal processing communities has led to new tools to process and analyze time-varying graphs and signals on a graph, such as multilayer graphs and tensor products of graphs [25]–[27]. The notion of joint time-vertex harmonic analysis was further realized in [19] by one of the authors of this work. Therein, the joint Fourier analysis is presented and its properties analyzed in details, together with examples of joint filters. In this work, we leverage these concepts proposing a framework in which the joint Fourier transform is just one of the building blocks.

Visualization, filtering and stationarity. The idea of analyzing the behavior of graph filters with time-varying signals first appeared in [28], showing that graph filters could be analyzed by applying jointly a GFT and a Z-transform and as such they possess a joint frequency response. Since then, we have seen a number of works dealing with time-varying signals on graphs: Authors in [29] propose a method that relies on graph wavelet theory and product graphs to visualize time-varying data defined on the vertices of a graph in order to identify spatial and/or temporal variations. A step towards the graphical model has been carried out by authors in [30]. In this work, authors assume data time dependencies to be modeled by an auto-regressive (AR) process and they propose several algorithms to estimate the network structure capturing the spatio-temporal dependencies and the coefficients of the AR process expressed as graph polynomial filters. In order to deal with the high computational complexity of the eigendecomposition, different filtering approximation algorithms have been proposed, mainly based on polynomials: centralized and distributed joint filter 2D Chebychev polynomial [19], separable rational [20] implementations, and autoregressive models [31].

Finally, in parallel with this work, the authors extended the notions of time stationarity and the recent graph stationarity [12] to the joint time-vertex domain [32] providing a framework for the statistical signal processing of time-vertex signals. Authors showed that assuming joint stationarity to regularize learning can yield significant accuracy improvements and reduce computational complexity in both estimation and recovery tasks prediction with respect to purely time or graph methods [33], [34]. Despite the relevance of this work to time-vertex analysis, here we focus on the purely deterministic setting.

II. Harmonic Time-Vertex Analysis

We denote by $G = (\mathcal{V}, \mathcal{E}, \boldsymbol{W}_G)$ the graph, where $\mathcal{V}$ indicates the set of nodes, $\mathcal{E}$ the set of edges and $\boldsymbol{W}_G$ is the associated $N \times N$ weight matrix. Furthermore, let $\boldsymbol{L}_G = \boldsymbol{D}_G - \boldsymbol{W}_G$ be the combinatorial Laplacian matrix, i.e. the finite difference approximation to the continuous Laplacian operator [17] or the Laplace-Beltrami operator for Riemannian manifolds [35]. We suppose that the signal on a graph is sampled at T successive regular intervals of unit length. That is, if we denote by $\boldsymbol{x}_t \in \mathbb{R}^N$ the graph signal at instant t, the time-varying graph signal corresponds to the matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T] \in \mathbb{R}^{N \times T}$. We denote $\boldsymbol{X}^\intercal$, $\bar{\boldsymbol{X}}$, and $\boldsymbol{X}^*$ the transpose, the complex conjugate and the

hermitian of $\boldsymbol{X}$. Furthermore, we refer to both $\boldsymbol{X}$ and its vectorized form $\boldsymbol{x} = \mathrm{vec}(\boldsymbol{X}) \in \mathbb{R}^{NT}$ as "time-vertex signal".

A. The joint time-vertex Fourier transform

The main idea of harmonic analysis is to decompose a signal into oscillating modes thanks to the Fourier transform. For instance, one analyses oscillations along the temporal axis by applying the Discrete Fourier Transform (DFT) independently to each row of $\boldsymbol{X}$

$$\mathrm{DFT}\{\boldsymbol{X}\} = \boldsymbol{X}\bar{\boldsymbol{U}}_T, \tag{1}$$

where $\boldsymbol{U}_T$ is the normalized DFT matrix defined as

$$\boldsymbol{U}_T^*(t,k) = \frac{e^{-j\omega_k t}}{\sqrt{T}}, \quad \text{with} \quad \omega_k = \frac{2\pi(k-1)}{T}, \tag{2}$$

with $t, k = 1, 2, \ldots, T$. Similarly, the Graph Fourier Transform (GFT) [8], [14], [36] allows us to analyze oscillations along the graph edges. As each column of $\boldsymbol{X}$ represents a time instant, the GFT of $\boldsymbol{X}$ for all t reads

$$\mathrm{GFT}\{\boldsymbol{X}\} = \widetilde{\boldsymbol{X}} = \boldsymbol{U}_G^*\boldsymbol{X}, \tag{3}$$

where $\boldsymbol{U}_G$ is obtained by the eigendecomposition $\boldsymbol{L}_G = \boldsymbol{U}_G\boldsymbol{\Lambda}_G\boldsymbol{U}_G^*$ of the graph Laplacian. As $\boldsymbol{U}_G$ is orthonormal, the inverse Fourier transform becomes $\mathrm{GFT}^{-1}\{\widetilde{\boldsymbol{X}}\} = \boldsymbol{X} = \boldsymbol{U}_G\widetilde{\boldsymbol{X}}$. This spectral decomposition gives rise to a graph-specific notion of frequency as their squared modulus corresponds the Laplacian eigenvalue $\boldsymbol{\Lambda}_G(\ell,\ell) = \lambda_\ell$.

Harmonic time-vertex analysis amounts to analyzing oscillations *jointly* along both the time and the vertex dimensions. Hence, assuming a non-varying graph in time, the *joint time-vertex Fourier transform*, or JFT for short, is obtained by applying the GFT on the graph dimension and the DFT along the time dimension [19]

$$\widehat{\boldsymbol{X}}(\ell,k) = \frac{1}{\sqrt{T}}\sum_{n=1}^{N}\sum_{t=1}^{T}\boldsymbol{X}(n,t)\boldsymbol{u}_\ell^*(n)e^{-j\omega_k t}.$$

The above expression can be conveniently rewritten in matrix form as

$$\widehat{\boldsymbol{X}} = \mathrm{JFT}\{\boldsymbol{X}\} = \boldsymbol{U}_G^*\boldsymbol{X}\overline{\boldsymbol{U}}_T. \tag{4}$$

Expressed in vector form, the transform becomes

$$\hat{\boldsymbol{x}} = \mathrm{JFT}\{\boldsymbol{x}\} = \boldsymbol{U}_J^*\boldsymbol{x}, \tag{5}$$

where $\boldsymbol{U}_J = \boldsymbol{U}_T \otimes \boldsymbol{U}_G$ is the Kronecker product of the basis. The relation between Eq.(4) and Eq.(5) is obtained through the property of the Kronecker product $(\boldsymbol{M}_1 \otimes \boldsymbol{M}_2)\,\boldsymbol{x} = \boldsymbol{M}_2\boldsymbol{X}\boldsymbol{M}_1^\intercal$.

Properties of JFT.

Property 1. *JFT is an invertible transform. The inverse JFT in matrix and vector form are, respectively,* $\mathrm{JFT}^{-1}\{\widehat{\boldsymbol{X}}\} = \boldsymbol{U}_G\boldsymbol{X}\boldsymbol{U}_T^\intercal$ *and* $\mathrm{JFT}^{-1}\{\hat{\boldsymbol{x}}\} = \boldsymbol{U}_J\boldsymbol{x}$.

Property 2. *The Parseval relation holds:*

$$\sum_{n,t=1}^{N,T}|\boldsymbol{X}(n,t)|^2 = \sum_{\ell,k=1}^{N,T}|\widehat{\boldsymbol{X}}(\ell,k)|^2. \tag{6}$$

Property 3. *The transform is independent on the order GFT and DFT are applied to the time-vertex signal*

$$\mathrm{JFT}\{\boldsymbol{X}\} = \mathrm{GFT}\{\mathrm{DFT}\{\boldsymbol{X}\}\} = \mathrm{DFT}\{\mathrm{GFT}\{\boldsymbol{X}\}\}.$$

Property 4. *The subspace of zero graph and temporal frequency is spanned by the constant time-vertex signal* $\mathbf{1}\mathbf{1}^*$*, with* $\mathbf{1}$ *the all-ones vector.*

B. Time-vertex calculus and variation

In the following, we briefly present the main time-vertex differential operators. These will help us (a) to perform calculus on a finite, discrete time and space, and (b) to characterize the properties of the signals, such as smoothness, while taking into account the intrinsic structure of the data domain.

Time and vertex domains. Before introducing the time-vertex operators, we momentarily diverge by presenting the standard definitions in the time and graph domains. The main discrete calculus operator in time is the first order difference operator

$$\boldsymbol{X}\nabla_T|_t = \boldsymbol{x}_t - \boldsymbol{x}_{t-1},$$

taken here with periodic boundary conditions. Hence, the symmetric time Laplacian matrix $\boldsymbol{L}_T = \nabla_T^*\nabla_T$ is the discrete second order derivative in time with reversed sign

$$\boldsymbol{X}\boldsymbol{L}_T|_t = -\boldsymbol{x}_{t+1} + 2\boldsymbol{x}_t - \boldsymbol{x}_{t-1} \tag{7}$$

with $\boldsymbol{x}_{t+1} = \boldsymbol{x}_1$. As a circulant matrix, it has eigendecomposition $\boldsymbol{L}_T = \boldsymbol{U}_T\boldsymbol{\Lambda}_T\boldsymbol{U}_T^*$, where

$$\boldsymbol{\Lambda}_T(k,k) = 2\left(1 - \cos\left(\omega_k\right)\right). \tag{8}$$

The operator corresponding to the time derivative in the vertex is the edge derivative. Given a graph signal $\boldsymbol{x} \in \mathbb{R}^N$, the edge derivative with respect to edge $e = (n,m)$ at vertex n is given by

$$\left.\frac{\partial \boldsymbol{X}}{\partial e}\right|_n = \sqrt{\boldsymbol{W}(n,m)}\left[\boldsymbol{x}^n - \boldsymbol{x}^m\right]. \tag{9}$$

Therefore the graph gradient of $\boldsymbol{x}$ at vertex n is

$$\nabla_G\boldsymbol{x}|_n = \left\{\left.\frac{\partial \boldsymbol{x}}{\partial e}\right|_n\right\}_{e\in\mathcal{E}} \tag{10}$$

and, as before, $\boldsymbol{L}_G = \nabla_G^*\nabla_G$, where ∇_G^* is the divergence operator of the graph.

Joint domain. We define the joint gradient of a time-vertex signal $\boldsymbol{X}$ by concatenation of the time and graph gradients:

$$\nabla_J \boldsymbol{x} = \mathrm{vec}\left(\begin{bmatrix} \boldsymbol{X}\nabla_T^\intercal \\ \nabla_G \boldsymbol{X} \end{bmatrix} \right). \tag{11}$$

Therefore ∇_J can be rewritten as

$$\nabla_J = \begin{bmatrix} \nabla_T \otimes \boldsymbol{I}_G \\ \boldsymbol{I}_T \otimes \nabla_G \end{bmatrix}. \tag{12}$$

The Laplacian is classically defined to equal the divergence of the gradient, and also in our case the joint Laplacian is $\boldsymbol{L}_J = \nabla_J^* \nabla_J$. Expanding the expression while exploiting the mixed-product property of the Kronecker product, we find

$$\begin{aligned} \boldsymbol{L}_J &= (\nabla_T \otimes \boldsymbol{I}_G)^*(\nabla_T \otimes \boldsymbol{I}_G) + (\boldsymbol{I}_T \otimes \nabla_G)^*(\boldsymbol{I}_T \otimes \nabla_G) \\ &= (\nabla_T^* \otimes \boldsymbol{I}_G)(\nabla_T \otimes \boldsymbol{I}_G) + (\boldsymbol{I}_T \otimes \nabla_G^*)(\boldsymbol{I}_T \otimes \nabla_G) \\ &= (\nabla_T^* \nabla_T) \otimes \boldsymbol{I}_G + \boldsymbol{I}_T \otimes (\nabla_G^* \nabla_G) \\ &= \boldsymbol{L}_T \otimes \boldsymbol{I}_G + \boldsymbol{I}_T \otimes \boldsymbol{L}_G = \boldsymbol{L}_T \times \boldsymbol{L}_G, \end{aligned}$$

and therefore $\boldsymbol{L}_J$ is also equivalent to the Cartesian product between the time and graph Laplacians[1]. Above, and the second equality follows from the mixed-product property of the Kronecker product. Thus, the Laplacian operator $\boldsymbol{L}_J$ applied to the signal $\boldsymbol{x}$ is

$$\boldsymbol{L}_J \boldsymbol{x} = (\boldsymbol{L}_T \times \boldsymbol{L}_G)\, \boldsymbol{x} = \mathrm{vec}(\boldsymbol{L}_G \boldsymbol{X} + \boldsymbol{X} \boldsymbol{L}_T).$$

The result of the Cartesian product is a multilayer graph, referred to as the *joint graph* J, where the original graph G is copied at each time step $t = 1, 2, \ldots, T$. Additionally, each node at layer t is connected to itself at layer $t-1$ and $t+1$ with periodic boundary conditions. It is useful to remind that the Kronecker product of the two eigenvectors basis $\boldsymbol{U}_T$ and $\boldsymbol{U}_G$ diagonalize the joint Laplacian with eigenvalues equal to the sum of all the pairs (ω_k, λ_ℓ) [37]:

$$\begin{aligned} \boldsymbol{L}_J &= \boldsymbol{L}_T \otimes \boldsymbol{I}_G + \boldsymbol{I}_T \otimes \boldsymbol{L}_G \\ &= (\boldsymbol{U}_T \boldsymbol{\Lambda}_T \boldsymbol{U}_T^*) \otimes \boldsymbol{I}_G + \boldsymbol{I}_T \otimes (\boldsymbol{U}_G \boldsymbol{\Lambda}_G \boldsymbol{U}_G^*) \\ &= (\boldsymbol{U}_T \otimes \boldsymbol{U}_G)(\boldsymbol{\Lambda}_T \times \boldsymbol{\Lambda}_G)(\boldsymbol{U}_T \otimes \boldsymbol{U}_G)^* = \boldsymbol{U}_J \boldsymbol{\Lambda}_J \boldsymbol{U}_J^* \end{aligned}$$

where we have used the mixed-product property of the Kronecker product.

Measures of joint variation.

The gradient and its various norms are often used as regularizers in regression because they capture the variation of the signal over a domain of interest. The ℓ_2-norm of the joint gradient measures the total variation of the signal across edges and consecutive steps. Observe that

$$\begin{aligned} \|\nabla_J \boldsymbol{x}\|_2^2 &= \boldsymbol{x}^\intercal \boldsymbol{L}_J \boldsymbol{x} = \|\nabla_G \boldsymbol{X}\|_F^2 + \|\boldsymbol{X} \nabla_T\|_F^2 \\ &= \mathrm{tr}(\boldsymbol{X}^\intercal \boldsymbol{L}_G \boldsymbol{X}) + \mathrm{tr}(\boldsymbol{X} \boldsymbol{L}_T \boldsymbol{X}^\intercal) \end{aligned} \tag{13}$$

meaning that $\|\nabla_J \boldsymbol{x}\|_2^2$ is separable over the the two domains.

Analogously, the ℓ_1-norm of the joint gradient can be written as the sum of the ℓ_1-norms

$$\|\nabla_J \boldsymbol{x}\|_1 = \|\mathrm{vec}(\nabla_G \boldsymbol{X})\|_1 + \|\mathrm{vec}(\boldsymbol{X} \nabla_T)\|_1, \tag{14}$$

which is often referred as the Total Variation (TV) norm. In general, it is possible to define a mixed norm $\mathrm{N}_{p,q}(\cdot)$

$$\mathrm{N}_{p,q}(\boldsymbol{x}) \triangleq w_G \|\mathrm{vec}(\nabla_G \boldsymbol{X})\|_p^p + w_T \|\mathrm{vec}(\boldsymbol{X} \nabla_T)\|_q^q \tag{15}$$

where the p-norm and the q-norm are computed independently on the two domains and w_G, w_T are non-negative weights. Such norms are often useful when the signal vary differently (e.g., smooth or piece-wise) across the two domains, as we will show in Section VI-B.

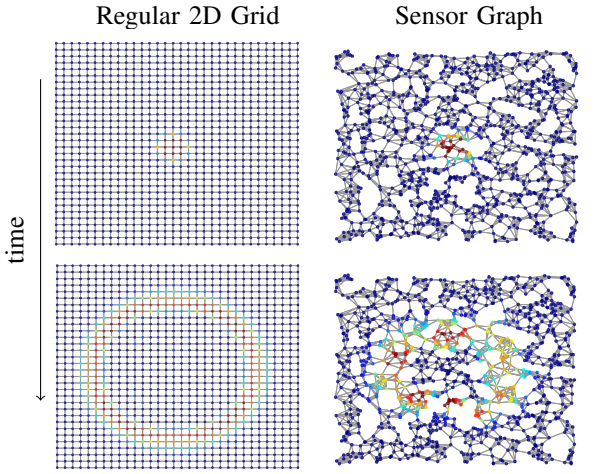

Fig. 1. Solution to the wave equation on a regular 2D grid and on a sensor graph at different point in time. The propagating behavior is evident even in the case of irregular domain.

III. Dynamics over Graphs

This section motivates the JFT further by showing that it can be used to characterize two linear PDEs evolving over the graph by kernels defined in the joint frequency domain, and also to provide insight on standard non-linear PDEs used in epidemic modeling. Our interest in PDEs analysis is related to their capability of encoding information about the structure of the underlying domain, whether continuous or discrete [38], [39] Moreover, PDEs are not only simple but powerful models of many phenomena observed in reality, but also a motivation for the Fourier transform [40], [41].

[1]In this work we consider the Cartesian product for its amenable spectral properties, but in general other graph products could be considered, such as the Kronecker product $J = G \otimes G_T$ or the strong product $J = G \boxtimes G_T = G \times G_T + G \otimes G_T$ [24].

A. Linear dynamics on graphs

We are interested in linear PDEs whose solution at each time step can be expressed as a linear operator applied to the initial condition. In particular, we consider the heat diffusion and the wave equations defined in the discrete setting. We denote $\boldsymbol{x}_1$ the initial condition of the PDEs, or equivalently in the joint spectral domain $\boldsymbol{Z}(\ell, k) = \widetilde{\boldsymbol{x}_1}(\ell)\, \boldsymbol{U}_T^*(k, 1)$.

Heat equation. The discrete heat diffusion equation $\boldsymbol{x}_t - \boldsymbol{x}_{t-1} = -s\boldsymbol{L}_G\boldsymbol{x}_t$ is, arguably, one of the simplest dynamics described by differential equations. The parameter s represents thermal diffusivity and is interpreted as a scale parameter for multiscale dynamic graph wavelet analysis [15] and graph scale-space theory [42]. It is well understood that

$$\boldsymbol{x}_t = (\boldsymbol{I} - s\boldsymbol{L}_G)^{t-1}\boldsymbol{x}_1. \tag{16}$$

Evaluating both the GFT and DFT, one also finds that the solution also has distinct structure in the joint spectral domain

$$\widehat{\boldsymbol{X}}(\ell, k) = \frac{1}{\sqrt{T}} \frac{a(\lambda_\ell, \omega_k)^T - 1}{a(\lambda_\ell, \omega_k) - 1} \boldsymbol{Z}(\ell, k) \tag{17}$$

where $a(\lambda_\ell, \omega_k) = (1 - s\lambda_\ell)\, e^{-j\omega_k}$. The JFT of a heat diffusion process therefore exhibits a smooth non-separable low-pass form.

Wave equation More interesting dynamics can be modeled by the discrete second order differential equation

$$\boldsymbol{X}\boldsymbol{L}_T = s\boldsymbol{L}_G\boldsymbol{X}, \tag{18}$$

representing a discrete wave propagating on a graph with speed $s > 0$. Figure 1 shows the evolution of a signal obtained by solving (18) using a numerical iterative scheme on a regular 2D lattice and on a random sensor graph. It is clear that, assuming a sufficiently regular graph, the solution to Eq. (18) evolves on the graph as a wave propagates in the Euclidean domain. In the appendix we prove that, for vanishing initial velocity, the solution in the joint spectral domain can be written as

$$\widehat{\boldsymbol{X}}(\ell, k) = \widehat{K}_s(\lambda_\ell, \omega_k)\boldsymbol{Z}(\ell, k)$$

where

$$\widehat{K}_s(\lambda_\ell, \omega_k) = \sum_t \cos(t\theta_\ell) e^{-j\omega_k t} \tag{19}$$

and $\theta_\ell = \arccos(1 - \frac{s\lambda_\ell}{2})$. Since the $\arccos(x)$ is defined only for $x \in [-1, 1]$, to guarantee stability the parameter s must satisfy $s < 4/\lambda_{max}$.

The distinctive pattern of the JFT of a wave shown in Fig. 2 (bottom right) is sparser and more structured that the original (top left), GFT (top right), and DFT (bottom left) representations.

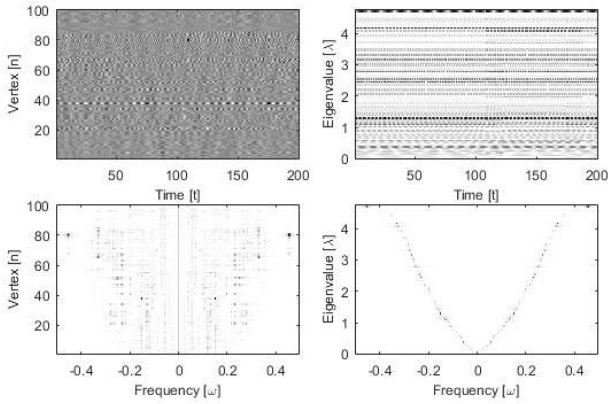

Fig. 2. Frequency analysis of multiple waves propagating on a random sensor graph. The signals on each node of the graph evolve according to a PDE, but their time-vertex representation (top left) does not highlight any relation between the two domains. Similarly, GFT (top right) and DFT (bottom left) are not able to show the underlying structure. It is evident that JFT (bottom right) succeeds in representing the signal in the most meaningful way, revealing its regular pattern.

B. Complex dynamics over networks: the illustrative example of epidemic models

Time-vertex harmonic analysis often provides useful insights on the dynamics of a signal even when the latter is not characterized by a linear PDE. To illustrate this, we will show how the JFT can be used to characterize the evolution of a non-linear, discrete, and non-deterministic model for the spread of an infectious disease.

In particular, we focus on the dynamics corresponding to different compartmental models commonly used in epidemiology. We simulated the epidemics spreading over N = 695 cities of Europe according to two different models: the Susceptible-Exposed-Infected-Recovered (SEIR) model and the SEIRS model, where the immunity of recovered individuals is only temporary. The models are parametrized by the contagion probability of infection, the infectious, latent and immunity periods and the population per city. Each node of the graph represents a city with a fixed population of individuals. Connections within the cities are modeled using randomly generated Erdős-Rényi graphs. Inter-cities connection are modeled using two graphs, a terrestrial coordinate-based graph and the graph of airline connections between the major city in Europe.

We simulate different realizations of the epidemic breakout, varying the parameters and observing how they influence the joint spectrum of the signals describing the evolution of the number of infected individuals at each node. Figure 3 shows the JFT of those signals. In the first case, the spectrum is characterized by regularly spaced lines along the angular frequency axis, caused by the inherent periodicity of the SEIRS model, where every individual can be infected again after the temporary

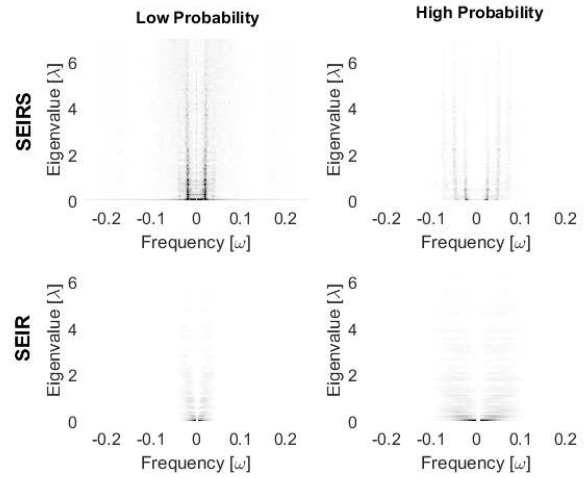

Fig. 3. JFT for different realization of epidemic spreads, modeled using different models and contagion probability. The transform allows us to distinguish between the different parameters of the model.

immunity period ceases. The spacing between the lines is due to both the immunity and latent periods. In the second case the epidemic has a more diffusive behaviour, without evident periodicity. For each one of the models, we simulate scenarios with high and low probabilities of contagion. It can be observed that the high probability case is characterized by a larger bandwidth occupancy with respect to the low probability one, due to a more impulsive behaviour of the epidemic breakout. Therefore, JFT seems to reveal the underlying structure and allows us to differentiate between the different models.

IV. Fast Filtering of Time-Vertex Signals

After recalling the definition of joint filters, we next present a novel algorithm to perform fast filtering on large graphs. Experiments illustrate that our algorithm achieves significantly better approximation of filtering objectives than state-of-the-art, while also not being constrained to a specific class of separable responses.

A. Joint time-vertex filters

A joint filter $h(\boldsymbol{L}_G, \boldsymbol{L}_T)$ is a function defined in the joint two-dimensional spectral domain $h : \mathbb{R}_+ \times \mathbb{R} \mapsto \mathbb{C}$ and is evaluated at the graph eigenvalues λ and the angular frequencies ω. Similarly to both the classical and the graph case, the generalized filtering operator is applied by means of the convolution theorem, i.e., point-wise multiplication in the spectral domain. The output of a joint filter is

$$h(\boldsymbol{L}_G, \boldsymbol{L}_T)\, \boldsymbol{x} = \boldsymbol{U}_J\, h(\boldsymbol{\Lambda}_G, \boldsymbol{\Omega})\, \boldsymbol{U}_J^* \boldsymbol{x}, \qquad (20)$$

Fig. 4. The effect of joint filters is easily visualized for the case of a dynamic mesh of a dancer. By filtering the original mesh (left) using a joint low-pass separable filter one approximates the time-varying skeleton of the dancer (center). Using a non-separable wave filter, the fluidity of the dancer's motion is emphasized (right).

where $h(\boldsymbol{\Lambda}_G, \boldsymbol{\Omega})$ is a diagonal $NT \times NT$ matrix defined as

$$h(\boldsymbol{\Lambda}_G, \boldsymbol{\Omega}) = \mathrm{diag}\left(\begin{bmatrix} h(\lambda_1, \omega_1) & \cdots & h(\lambda_1, \omega_T) \\ \vdots & \ddots & \vdots \\ h(\lambda_N, \omega_1) & \cdots & h(\lambda_N, \omega_T) \end{bmatrix} \right)$$

and the $\mathrm{diag}(\boldsymbol{A})$ operator creates a matrix with diagonal elements the vectorized form of $\boldsymbol{A}$.

Illustration: dynamic mesh filtering. Figure 4 shows an example of joint filtering of a mesh representing a dancer[2]. We design (a) a joint separable lowpass filter that attenuates high frequency components in both graph and time domains, and (b) a wave filter whose frequency response is described in Eq. (19). In the first case, we obtain the approximate skeleton of the mesh with rigid movements, whereas the wave filter produce a fluid (wavy) dancer, enhancing the frequency components in a non-linear fashion. We remark that this effect can only be obtained using non-separable filters.

Separable vs. non-separable filters. A notable family of joint filters are those that have *separable* response

$$h(\lambda, \omega) = h_1(\lambda)\, h_2(\omega). \qquad (21)$$

These filters have a straightforward interpretation: the frequency response of each filter affects only the domain where it is defined

$$h(\boldsymbol{L}_G, \boldsymbol{L}_T)\boldsymbol{x} = \mathrm{vec}(h_1(\boldsymbol{L}_G)\boldsymbol{X} h_2(\boldsymbol{L}_T))\,. \qquad (22)$$

Moreover, since they can be designed independently at the two domains, joint filters can be obtained by combining graph and temporal filters [20]. However, due to their simple form, separable filters cannot model the dynamics of PDEs (e.g., waves or heat diffusion), where there is an interplay between the temporal and graph frequency domains. For this reason, in the following we aim to find an efficient joints filter implementation for separable as well as non-separable filtering objectives.

[2]http://research.microsoft.com/en-us/um/redmond/events/geometrycompression/data/default.html

B. Fast joint filtering

Due to the high complexity of eigendecomposition, graph filters are almost always implemented using fast 2D polynomial [43] and rational [31] approximations. In the context of time-vertex analysis, the importance of fast joint filtering is emphasized by the increase of the problem's dimensions. Recognizing this need, researchers have recently proposed distributed joint filter 2D Chebychev polynomial [19] and separable rational [20] implementations, appropriate for arbitrary and separable joint response functions, respectively. In the following, we improve upon state of the art by enhancing the filtering approximation at a similar (up to logarithmic factors) complexity.

The Fast Fourier Chebyshev (FFC) algorithm. The basic idea of our algorithm is to exploit the small complexity of FFT and perform graph filtering in the time-frequency domain. Concretely, to filter $\boldsymbol{X}$ with response $h(\lambda, \omega)$, we do the following:

1. Compute the FFT of every row of $\boldsymbol{X}$, at a total complexity of $\mathcal{O}(NT \log T)$.
2. For each ω_k, approximate $h(\lambda, \omega_k)$ with a Chebyshev polynomial of order M_G and use the fast graph Chebyshev recursion [14] to filter the corresponding angular frequency component of $\boldsymbol{X}$. The complexity of this step is $\mathcal{O}(M_G T|\mathcal{E}|)$.
3. Use the inverse FFT to obtain the filtered time-vertex signal, with complexity $\mathcal{O}(NT \log T)$.

Our scheme can approximate both separable or non-separable joint filters using $\mathcal{O}(T|\mathcal{E}|M_G + NT \log T)$ operations, which up to a logarithmic factor is a linear complexity to the number of edges $|\mathcal{E}|$, nodes N, timesteps T, and filter order M_G. Moreover, it can be performed distributedly since both the FFT and the graph Chebychev recursion necessitate only local or few hop information.

Numerical comparison. To evaluate the approximation properties of the above scheme, we show in Figure 5 numerical experiments for an ideal separable lowpass filter and a non-separable wave filter on a time-vertex graph with size $N = 5000$, $T = 3000$. In detail, the approximated filtering functions (low pass and wave) are, respectively,

$$h_{\mathrm{LP}}(\lambda_\ell, \omega_k) = \frac{e^{-(\lambda_\ell - \lambda_{\mathrm{cf}})}}{1 + e^{-(\lambda_\ell - \lambda_{\mathrm{cf}})}} \frac{e^{-(|\omega_k| - \omega_{\mathrm{cf}})}}{1 + e^{-(|\omega_k| - \omega_{\mathrm{cf}})}} \quad (23)$$

$$h_{\mathrm{wave}}(\lambda_\ell, \omega_k) = e^{-|\pi|\omega_k| - \arccos(1 - \lambda_\ell/(2\lambda_{\max})|^2}. \quad (24)$$

For each case, we compare our algorithm with the state-of-the-art, i.e., Chebyshev2D approximation [19] of complexity and the ARMA2D approach [20], while choosing M_G and M_T as graph and temporal polynomial orders, respectively (here $M_G = M_T$).

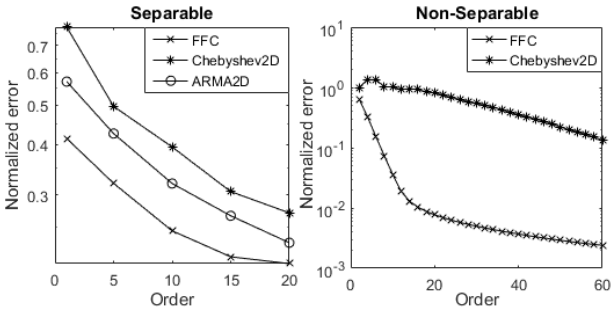

Fig. 5. Fast joint filtering comparison using different algorithms to approximate the ideal joint lowpass filter (left) and a non-separable wave filter (right) approximated in Eq (23) and Eq (24), respectively. The proposed method (FFC) outperforms the others, in particular for non-separable filters.

As shown in Figure 5, FFC results in a significant improvement in accuracy for the same order and the difference is particularly prominent in the non-separable case (ARMA2D cannot be used here). We remark however that, to interpret these results correctly, one has to consider the complexity of each method:

method	complexity	applicability
FFC	$\mathcal{O}(T\|\mathcal{E}\|M_G + NT \log T)$	all
Cheby2D [19]	$\mathcal{O}(T\|\mathcal{E}\|M_T + NTM_TM_G)$	all
ARMA2D [20]	$\mathcal{O}(T\|\mathcal{E}\|M_G + T\|\mathcal{E}\|M_T)$	separable

Therefore, for the same order, the three different methods feature slightly different complexities, implying that a direct comparison of accuracy is not entirely fair. Nevertheless, the unfairness in not in our favor as, in our experiments for all orders larger than 2, the asymptotic complexity of FFC is the smallest (since here $M_G = M_T$, $\log T < M_T M_G$, and $\log T < |\mathcal{E}|M_T/N$). We also note that, in practice one often needs $M_T \gg \log T$ to achieve a good approximation, in which case FFC is the fastest.

V. Time-vertex Dictionaries and Frames

So far, we have looked at time-vertex signals through the lenses of the canonical and the joint Fourier bases. However, in some cases it is beneficial to also consider alternative representations. For example, in the classic case, the wavelet and the short time Fourier transforms respectively enable time-scale and time-frequency analysis of the signal. The purpose of this section is to show how one can define analogous representations for time-vertex signals. These can be used for instance to generate features given as an input to a classifier (see Section VI-C) or to regularize an optimization problem such as (41) in Section VI-C.

Classically, the atoms of the representations are built by applying a transform (e.g. scaling or modulation) to a mother function and shifting the resulting functions. We follow a similar approach, with the difference that the mother function is replaced by a kernel defined in the time-vertex frequency domain and the shifting has to be replaced by an operator suitable to graphs. The spectral time-vertex wavelet and the short time vertex Fourier transforms follow as consequences of our framework.

A. Joint time-vertex localization

The ability to localize a kernel over a particular time and vertex is a key ingredient of our dictionary construction. In the following, we derive such a joint localization operator as a generalization of the graph localization operator [12], [36], [44], which localizes a kernel $h(\boldsymbol{L}_G)$ onto vertex v_m

$$\mathcal{T}_m^G h \triangleq h(\boldsymbol{L}_G)\,\boldsymbol{\delta}_m = \sum_{\ell=1}^{N} h(\lambda_\ell)\,\bar{\boldsymbol{u}}_\ell(m)\,\boldsymbol{u}_\ell. \tag{25}$$

Above, $\boldsymbol{\delta}_m$ is a Kronecker delta centered at vertex v_m. Similarly, in the joint domain we define the *joint time-vertex localization operator* as the filtering with a two-dimensional Kronecker delta

$$\mathcal{T}_{m,\tau}^J h \triangleq h(\boldsymbol{L}_G, \boldsymbol{L}_T)\,(\boldsymbol{\delta}_m \otimes \boldsymbol{\delta}_\tau)\,. \tag{26}$$

It turns out that the joint time-vertex localization operator has the advantages of both the graph localization and the traditional translation operator. Indeed, we observe the following relations

$$\begin{aligned}\mathcal{T}_{m,\tau}^J h(n,t) &= \frac{1}{T}\sum_{\substack{\ell=1\\k=1}}^{N,T} h(\lambda_\ell,\omega_k)\bar{\boldsymbol{u}}_\ell(m)e^{-j\omega_k\tau}\boldsymbol{u}_\ell(n)e^{j\omega_k t}\\ &= \frac{1}{T}\sum_{k=1}^{T}\left(\sum_{\ell=1}^{N} h(\lambda_\ell,\omega_k)\bar{\boldsymbol{u}}_\ell(m)\boldsymbol{u}_\ell(n)\right)e^{j\omega_k(t-\tau)}\end{aligned} \tag{27}$$

From (27), it follows that joint localization consists of three steps: (a) localizing independently all kernels $h(\cdot,\omega_k)$, (b) computing the inverse DFT along the other dimension, and (c) translating the result. Joint localization is thus equivalent to independent application of a graph localization and a translation. Note that the steps (a) and (b) can be considered as localizing the signal along the time dimension.

When the filter is separable, i.e. $h(\lambda,\omega) = h_G(\lambda)h_T(\omega)$, the joint localization is simply

$$h(\boldsymbol{L}_G, \boldsymbol{L}_T)\,(\boldsymbol{\delta}_\tau \otimes \boldsymbol{\delta}_m) = \mathrm{vec}(h_G(\boldsymbol{L}_G)(\boldsymbol{\delta}_\tau \otimes \boldsymbol{\delta}_m^\intercal)h_T(\boldsymbol{L}_T))\,, \tag{28}$$

showing that the filter can be localized independently in time and in the vertex domain.

B. Joint time-vertex dictionaries

We proceed to present our dictionary construction for time-vertex signals. We start with a mother time-vertex kernel $h(\lambda,\omega)$ and a transformation $s_{z_\lambda,z_\omega}(\cdot,\cdot)$ parametrized by some values $[z_\lambda, z_\omega]$ belonging to the finite 2D set $\mathcal{Z}_\lambda \times \mathcal{Z}_\omega \subset \mathbb{R}^2$ and controlling the kernel's shape along the vertex and time domains. The transformed kernel is then obtained by composition

$$h_{z_\lambda,z_\omega}(\lambda,\omega) = h(s_{z_\lambda,z_\omega}(\lambda,\omega)). \tag{29}$$

We build our dictionary by transforming $h(\lambda,\omega)$ with all $z_\lambda, z_\omega \in \mathcal{Z}_\lambda \times \mathcal{Z}_\omega$, (possibly) normalizing, and jointly localizing the resulting kernels $h_{z_\lambda,z_\omega}(\lambda,\omega)$ at each node m and time τ. Concretely, the dictionary is

$$\begin{aligned}\mathcal{D}_h = \{\mathcal{T}_{m,\tau}^J h_{z_\lambda,z_\omega}\} \quad &\text{for} \quad m \in \mathcal{V},\ \tau = 1,2,\ldots,T,\\ &\text{and } [z_\lambda, z_\omega] \in \mathcal{Z}_\lambda \times \mathcal{Z}_\omega\}.\end{aligned} \tag{30}$$

When $\mathcal{D}_h$ is overly redundant, one may choose to consider only a subset of values for m and τ.

We next consider two interesting examples of the proposed dictionary construction that are generalizations of the short-time Fourier and wavelet transforms [45], [46]:

Short Time-Vertex Fourier Transform (STVFT). Set $s_{z_\lambda,z_\omega}(\cdot,\cdot)$ to a shift in the spectral domain

$$s_{z_\lambda,z_\omega}(\lambda,\omega) = [\lambda - z_\lambda, \omega - z_\omega]. \tag{31}$$

This transform can be considered as a modulation. Nevertheless, we note that it does not correspond to a multiplication by an eigenvector as in [36], [47]. Our construction is more related to [48, Section 3]. Then, given a separable mother kernel $h(\lambda,\omega) = h_G(\lambda)\,h_T(\omega)$ and a finite 2D set $\mathcal{Z}_\lambda \times \mathcal{Z}_\omega \subset \mathbb{R}^2$, the STVFT of signal $\boldsymbol{X}$ is defined as

$$\begin{aligned}\mathrm{STVFT}\{\boldsymbol{X}\}(m,\tau,z_\lambda,z_\omega) &\triangleq \langle \boldsymbol{X}, \mathcal{T}_{m,\tau}^J h(\lambda - z_\lambda, \omega - z_\omega)\rangle\\ &= \frac{1}{\sqrt{T}}\sum_{\ell,k} h(\lambda_\ell - z_\lambda, \omega_k - z_\omega)\widehat{\boldsymbol{X}}(\ell,k)\boldsymbol{u}_\ell(m)e^{j\omega_k\tau}.\end{aligned}$$

Provided that $h(\lambda,\omega)$ is localized around $[0,0]$, the amplitude of the coefficient (n,t,z_λ,z_ω) indicates the presence of the spectral mode $[z_\lambda, z_\omega]$ at vertex m and time τ. Moreover, since the mother kernel is separable, the design in the two domains can be performed independently:

For the graph domain, we suggest to select the values of z_λ to be equally spaced in $[0, \lambda_{\max}]$ [48, Section 3]. The spacing should be selected such that $\sum_{z_\lambda \in \mathcal{Z}_\lambda} h_G^2(\lambda_\ell - z_\lambda) \approx c$ for every λ_ℓ, ensuring good conditioning of the associated frame (see [14, Theorem 5.6]). Because of the graph irregularity, in most of the cases, we need to keep all possible values for m, i.e., $m = 1,2,\ldots,N$.

For the time domain, we recover a traditional STFT, with the difference that $h_T(\omega)$ is defined in the spectral domain. Nevertheless, for convenience, the window can still be designed in the time domain. As a rule of thumb $|\mathcal{Z}_\omega| = l_{h_T}$, where l_{h_T} is the support of h_T in the time domain[3], and the values of τ should be sampled regularly with a spacing $\frac{l_{h_T}}{R}$, where R is the desired redundancy. For a more complete treatment we refer the reader to [45].

Spectral Time-Vertex Wavelet Tranform (STVWT). Following the idea developed in [14], we set $s_{z_\lambda,z_\omega}(\cdot,\cdot)$ to a generalized graph dilation (or scaling), i.e., a multiplication in the spectral domain

$$s_{z_\lambda,z_\omega}(\lambda,\omega) = [z_\lambda\,\lambda,\, z_\omega\,\omega]. \tag{32}$$

Then, given a kernel $h(\lambda,\omega)$ the STVWT of $\boldsymbol{X}$ reads

$$\begin{aligned}\mathrm{STVWT}\{\boldsymbol{X}\}(m,\tau,z_\lambda,z_\omega) &\triangleq \langle \boldsymbol{X}, \mathcal{T}^J_{m,\tau}\, h(z_\lambda\,\lambda, z_\omega\,\omega)\rangle \\ &= \frac{1}{\sqrt{T}}\sum_{\ell,k} h(z_\lambda\,\lambda, z_\omega\,\omega)\widehat{\boldsymbol{X}}(\ell,k)\boldsymbol{u}_\ell(m)e^{j\omega_k\tau},\end{aligned}$$

where z_λ, z_ω are the scale parameters for the vertex and the time dimensions. A usual requirement for $h(\lambda,\omega)$ is that it has a zero DC component, i.e., $h(0,0) = 0$. Contrarily to the STVFT, the mother kernel here may not be separable, as illustrated in VI-C. The choice of the discretization lattice $[m,\tau,z_\lambda,z_\omega]$ is thus more involved and case dependent: we suggest that m and τ take all possible N and T values respectively, while z_λ and z_ω are carefully selected depending on the application. This choice is justified by the computational complexity detailed in the following.

C. Joint time-vertex frames

To make the proposed dictionaries and associated signal representations usable in practice, we next provide answers to three key questions: (a) How can we compute the representations efficiently (i.e., performing analysis and synthesis)? (b) How can we guarantee that the associated transforms are well conditioned such that they can be successfully inverted? (c) How to efficiently invert them, recovering the original signal? The second point is particularly important since a well conditioned transform allows for more robust representations, for instance when the dictionary is used to solve a synthesis or analysis regression problem with a sparse regularizer.

Efficient analysis and synthesis. The dictionary atoms can be seen as a filter-bank $\{h_z(\lambda,\omega)\}_{z\in\mathcal{Z}_\lambda\times\mathcal{Z}_\omega}$, in which case the operators going from the signal to the representation domain and back are the *analysis* operator

$$D_h\{\boldsymbol{X}\}(m,\tau,z) = \langle \boldsymbol{X}, \mathcal{T}^J_{m,\tau}\, h_z\rangle = \boldsymbol{C}_z(m,\tau),$$

and the *synthesis* operator

$$D^*_h\{\boldsymbol{C}\}(n,t) = \sum_z \langle \boldsymbol{C}_z, \mathcal{T}^J_{n,t}\, h_z\rangle = \boldsymbol{Y}(n,t).$$

Notice that in general $\boldsymbol{X} \neq \boldsymbol{Y}$, and equality holds only when the filter-bank is a unitary tight frame.

Instead of computing the dictionary explicitly (an operation that is costly both in memory and in computations), one may acquire the analysis coefficients for all m,τ by joint filtering $\boldsymbol{X}$ with kernel h_{z_λ,z_ω} taking advantage of the relation $\boldsymbol{C}_z = \mathrm{mat}(h_z(\boldsymbol{L}_G,\boldsymbol{L}_T)\boldsymbol{x})$. Similarly synthesis can be performed by summing filtering operations. Using our FFC filtering algorithm presented in Section IV-B, the total analysis complexity is thus $\mathcal{O}(|\mathcal{Z}_\lambda \times \mathcal{Z}_\omega|(T|\mathcal{E}|M_G + NT\log T))$, where typically $M_G \approx 50$.

The drawback of this technique is that it does not allow us to take advantage of sub-sampling the lattice $[n,t]$, especially in the non-separable case. Indeed, a filtering operation will always provide all coefficients regardless of the desired lattice. While addressing this computational problem is beyond the scope of this contribution, we note that we can still efficiently perform a sub-sampled STVFT by first filtering only in the graph domain and second computing a traditional STFT.

Conditioning and frame bounds. In several applications in signal processing one is interested not only in processing data in another convenient representation, but also to recover the original signal from its alternative representation. Redundant invertible dictionaries are referred to as *frames* [45], [49]. The following theorem generalizes the classic [45] results regarding the frame bounds, providing a condition for a joint time-vertex dictionary to be a frame, as in the case of graphs [48, Lemma 1], [14, Theorem 5.6].

Theorem 1. *Let $\{h_z(\lambda,\omega)\}_{z\in\mathcal{Z}_\lambda\times\mathcal{Z}_\omega}$ be the kernels of a time-vertex dictionary $\mathcal{D}_h$, and set*

$$A = \min_{l,k}\sum_z |h_z(\lambda_\ell,\omega_k)|^2,\ B = \max_{l,k}\sum_z |h_z(\lambda_\ell,\omega_k)|^2.$$

If $0 < A \leq B < \infty$, then D_h is a frame in the sense:

$$A\|\boldsymbol{X}\|_F^2 \leq \|D_h\{\boldsymbol{X}\}\|_F^2 \leq B\|\boldsymbol{X}\|_F^2 \tag{33}$$

for any time-vertex signal $\boldsymbol{X} \in \mathbb{R}^{N\times T}$.

The proof of theorem can be found in the appendix (see section A).

The theorem asserts that, if $A > 0$, no information is lost when the analysis operator is applied to a time-vertex signals, thus the transform is invertible. Furthermore, the

[3]In practice, the kernel is chosen to have a compact support in the time-domain.

ratio of the frame bounds A/B is related to the condition number of the frame operator $S_h\{\boldsymbol{X}\} = D_h^*\{D_h\{\boldsymbol{X}\}\}$, hence it is decisive for efficient reconstruction when we want to recover the signal from its representation solving an optimization problem [14, Section 7].

Efficient inversion. To recover the signal $\boldsymbol{X}$ from the coefficients $\boldsymbol{C}$, a solution is to use the pseudo-inverse, i.e. $\boldsymbol{X} = D_h^\dagger\{\boldsymbol{C}\}$ or to solve the following convex problem $\arg\min_{\boldsymbol{X}} \|D_h\{\boldsymbol{X}\} - \boldsymbol{C}\|_2^2$. Problematically, these are computationally intractable for large value of N and T. We will instead design a dual set of kernels that allows us to invert the transform by a single synthesis operation. To this end, we search for a set of filters $\tilde{h}$ such that $D_{\tilde{h}}^*\{D_h\{\boldsymbol{X}\}\} = \boldsymbol{X}$. It is not difficult to see that this equality is satisfied when

$$\sum_{z_\lambda, z_\omega} \tilde{h}_{z_\lambda, z_\omega}(\lambda_\ell, \omega_k)\, \overline{h_{z_\lambda, z_\omega}}(\lambda_\ell, \omega_k) = 1, \quad \forall\, \lambda_\ell, \omega_k. \tag{34}$$

Although redundant joint time-vertex frames admit an infinite number of dual kernel sets satisfying (34), the typical choice is to use the *canonical dual*, defined as

$$\tilde{h}_{z_\lambda, z_\omega}(\lambda_\ell, \omega_k) = \left(\sum_{z'_\lambda, z'_\omega} h^2_{z'_\lambda, z'_\omega}(\lambda_\ell, \omega_k) \right)^{-1} h_{z_\lambda, z_\omega}(\lambda_\ell, \omega_k). \tag{35}$$

In fact, this corresponds to the pseudo-inverse of D_h, i.e., $D_h^\dagger = D_{\tilde{h}}^*$, while also having a low computational complexity.

To summarize, given an invertible time-vertex transform D_h and coefficients $\boldsymbol{C}$, the inverse transform of D_h associated with the set of kernels $\{h_{z_\lambda, z_\omega}\}_{[z_\lambda, z_\omega] \in \mathcal{Z}}$ is

$$\boldsymbol{X} = D_{\tilde{h}}^*\{\boldsymbol{C}\} = \sum_{z_\lambda, z_\omega} \tilde{h}_{z_\lambda, z_\omega}(\boldsymbol{L}_G, \boldsymbol{L}_T) \boldsymbol{C}_{z_\lambda, z_\omega}, \tag{36}$$

where $\tilde{h}$ is defined in (35).

VI. Experiments

The suitability of the time-vertex framework for several classes of problems is illustrated on a wide variety of datasets: (a) dynamic meshes representing a walking dog and a dancing man, (b) the Caltrans Performance Measurement System (PeMS) traffic dataset depicting high resolution daily vehicle flow of 10 consecutive days in the highways of Sacramento measured every 5 minutes, (c) simulated SEIR- or SEIRS-type epidemics over Europe, (d) the Kuala Lumpur City Centre (KLCC) time-lapse video and (e) earthquake waveforms recorded by seismic stations geographically distributed in New Zealand, connected to the GeoNet Network.

Results suggest that joint analysis of time-vertex signals can bring forth benefits in signal denoising and recovery, learning and source localization problems. We remark that all the experiments were done using the GSPBOX [50], the UNLocBoX [51] and the LTFAT [52]. Code reproducing the experiments is available at https://lts2.epfl.ch/reproducible-research/a-time-vertex-signal-processing-framework/.

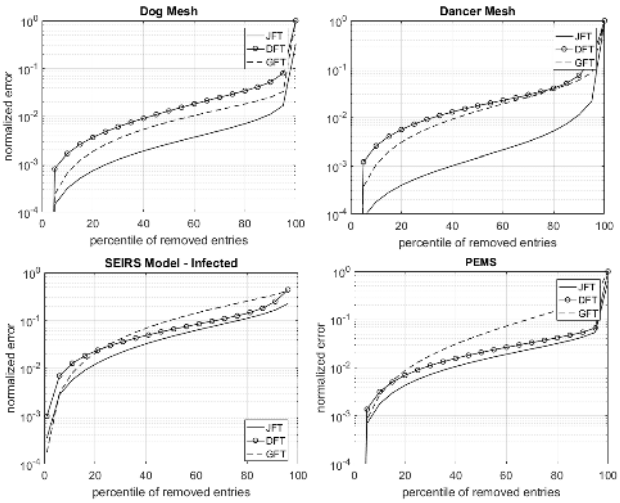

Fig. 6. Compactness of the transforms for different datasets: dog and dancer meshes (above) number of infected over Europe according to SEIRS model (below left) and traffic flow measured by the PeMS (below right). Normalized error is computed reconstruting the signal after thresholding the values of the transforms below the p-th percentile.

A. Compactness of representation

A key motivation behind the joint harmonic analysis is the capability of encoding time-varying graph-dependent signal evolution in a compact way. Our first step will therefore be to examine the energy compaction of the JFT transform in four datasets: two meshes representing a dancer ($N = 1502$ points in $\mathbb{R}^3$ and $T = 570$ timesteps) and a dog walking ($N = 2502$ points in $\mathbb{R}^3$ and $T = 52$ timesteps), the PeMS traffic flow dataset ($N = 710$ stations measuring traffic over $T = 2880$ intervals of 5-minute length each) and the number of infected individuals in an SEIRS epidemic (see Section III-B for description). Transforms with good energy compaction are desirable because they summarize the data well and can be used to construct efficient regularizers for regression problems.

To measure energy compactness, we compute the DFT, GFT and JFT for each dataset, we replace the spectrum coefficients with magnitudes smaller than the p-th percentile with zeros and perform the corresponding inverse transform on the resulting coefficients. Denoting by $\boldsymbol{X}$ the original signal and $\boldsymbol{X}_p$ the compressed one, the compression error is for each p given by $\|\boldsymbol{X}_p - \boldsymbol{X}\|_F / \|\boldsymbol{X}\|_F$. As shown in Figure 6, JFT exhibits

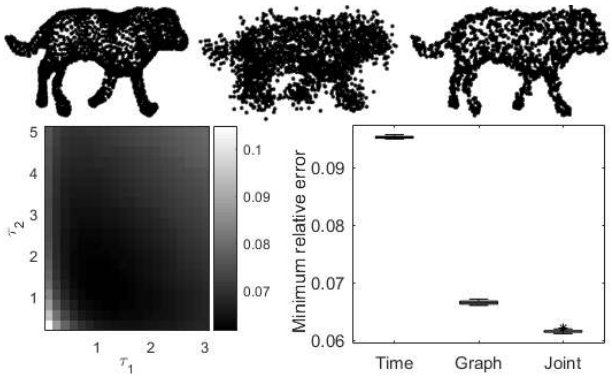

Fig. 7. Joint variation priors are useful in denoising the coordinates of dynamic meshes. The original mesh (left) was corrupted with random Gaussian noise (center, normalized error 0.2); shown here for one realization of the noise. After the denoising, the error decreases to 0.06 (right). The normalized error as a function of parameters τ_1 and τ_2 is shown in a heat-map (below left) averaged over 20 realizations. The boxplot (below right) shows the minimum achievable error for a time ($\tau = 0$), graph ($\tau_2 = 0$) and joint variation prior.

better energy compaction properties in all the datasets, and especially for the meshes where the graph captures well the signal structure.

B. Regression problems with joint variation priors

We next examine the utility of joint variation priors for regression problems in two example applications.

Denoising of dynamic meshes. Whenever a smoothness prior can be assumed, the joint Tikhonov regularization can be used to denoise a time-varying graph signal. The prior can be easily expressed in the time-vertex domain thanks to Eq. (13). Joint denoising is then performed by solving the following optimization problem

$$\arg\min_{\boldsymbol{X}} \|\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\tau_1\|\nabla_G\boldsymbol{X}\|_F^2+\tau_2\|\boldsymbol{X}\nabla_T\|_F^2, \quad (37)$$

where the regularization terms require the solution to be smooth in both graph and time domains. This problem has a closed form solution in our framework, which is a joint non-separable lowpass filter

$$h_{\mathrm{TIK}}(\lambda_\ell,\omega_k) = \frac{1}{1+\tau_1\lambda_\ell+2\tau_2(1-\cos(2\pi\omega_k))}. \quad (38)$$

In order to investigate its performance, we consider the vertices of a mesh of size $2502 \times 59 \times 3$ representing a walking dog, we add Gaussian noise to the coordinates, we build a k nearest neighbor graph based on the distances between the time average of the coordinates and finally we solve the problem (37) for each coordinate dimension. We averaged the results over 20 realizations of the noise.

The meshes in Figure 7 represent, from left to right, the original, the noisy and the recovered one, for one realization of the noise. Remarkably, the normalized error drops from 0.20 to 0.06, respectively before and after denoising, making the dog distinguishable again. As side effect, the dog appears to be thinner, due to the graph regularization. The heat-map in the left corner of the figure shows the role of the regularization parameters. We found (using exhaustive search) that the lowest error is achieved when $\tau_1 = 0.71$ and $\tau_2 = 1.78$. We compare the performance of the joint Tikhonov regularization with respect to time- and graph-only for the best parameter combinations of all methods. The boxplot on the right shows the minimum achievable error statistics in the three cases over the 20 realizations. It is easy to see that the graph plays a major role in the denoising, since it encodes the structural information of the mesh. Nevertheless, the joint approach performs the best, i.e., 0.062 ± 0.0002, taking advantage of the smoothness in both domains, while graph and time methods achieve 0.067 ± 0.0003 and 0.095 ± 0.0002, respectively.

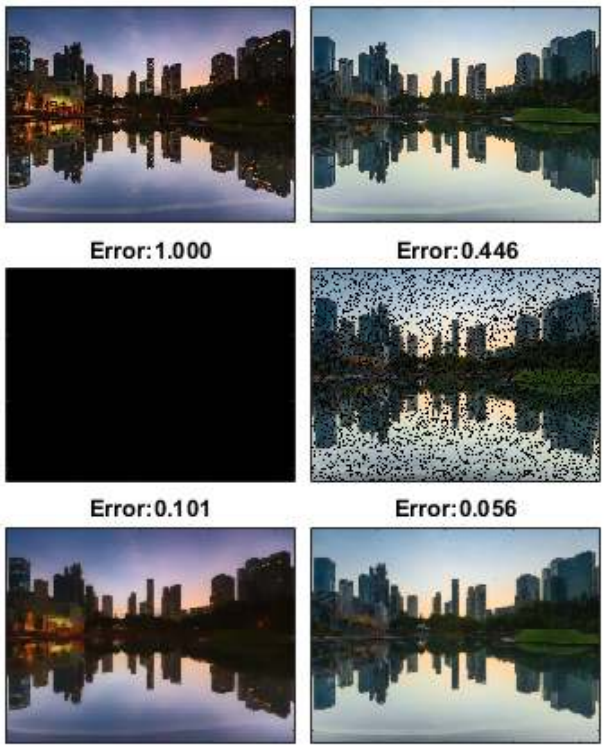

Fig. 8. Visual inspection of video inpainting results. We show two frames extracted from the video (top), their corrupted counterparts (middle left frame is missing entirely, whereas middle right frame has missing pixels), and the reconstructions using our method with an $N_{1,2}$ regularizer (bottom).

Inpainting of time-lapse video. We consider the problem of time-vertex signal recovery from noisy, corrupted, and incomplete measurements. Depending on the characteristics of the signal, the prior $\mathrm{N}_{p,q}(\boldsymbol{x})$ with

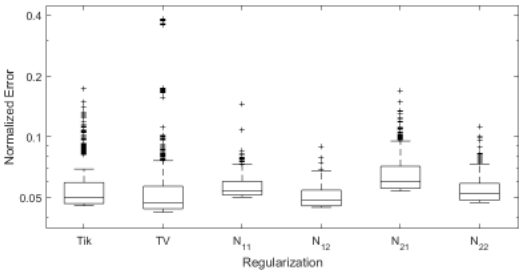

Fig. 9. Comparison of video inpainting performances between Tikhonov, TV and Joint regularizations. Each box represents a statistical summary of the error evaluated for each frame of the whole video. Although TV achieves the best recovery for some frames, in case of occluded frames the error is very large. Joint regularization $\mathrm{N}_{1,2}(\boldsymbol{x})$ trades the lowest error achievable with a better average recovery.

different values of p and q and different weights can be used. A typical signal recovery problem in signal processing is the image inpainting, i.e., trying to replace corrupted or lost part of the image. Since patch-graphs allow non-local image processing [53], our goal is to extend graph-based non-local processing to video inpainting and recovery. However, since our framework is constrained to static graphs, we focus on the particular case of time-lapse videos, whose structure stays majorly invariant throughout the video. To this end, we corrupted a time-lapse video that shows the skyline of the Kuala Lumpur City Centre, which statistical properties were amenable from a graph perspective, being the skyline static with time-varying colors. The video has size $160 \times 214 \times 3 \times 604$ (height $\times$ width $\times$ colors $\times$ frames). We removed 20% of the pixels and 20% of the frames from the original KLCC video, achieving a normalized error of 0.61. The inpainting is performed solving the optimization problem for each color using as regularizer $\mathrm{N}_{1,2}(\boldsymbol{x})$:

$$\arg\min_{\boldsymbol{X}} \|\boldsymbol{M} \circ \boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \gamma_1 \|\nabla_G \boldsymbol{X}\|_1 + \gamma_2 \|\boldsymbol{X} \nabla_T\|_F^2, \tag{39}$$

where $\boldsymbol{M}$ is the mask of the missing entries. The patch graph G is constructed from the video averaged in time. The rationale is that the l_1 over the graph will restore the missing pixels, being each frame approximately piecewise constant, whereas the l_2 norm in time recovers the smooth changes of the colors from dawn until dusk.

Figure 8 shows two frames of the video, their corrupted counterparts and the result of the recovery, along with the respective normalized errors. The recovered video has a normalized error of 0.049, illustrating that the joint inpainting is able to restore the global quality of the video even in case of considerable missing information.

We compare the recovery performance with all the joint regularizers $\mathrm{N}_{p,q}(\boldsymbol{x})$ for $p, q = \{1, 2\}$, and with two baseline algorithms, based on 3D-Tikhonov and isotropic 3D-TV regularizations [54]. The last two correspond to using a grid graph with equal weights on the edges. Table I reports the normalized errors averaged over the pixels-only, frames-only and the whole video. The better performance achieved by the joint regularizer $\mathrm{N}_{1,2}(\boldsymbol{x})$ is due to its capability to restore missing frames, while missing pixels recovery performances are almost the same. Figure 9 illustrates a summary statistics of the errors computed over each frame. Although TV performs the best in the median, in case of occluded frames the error is much larger w.r.t. the joint recovery, leading to a higher average error.

TABLE I
VIDEO INPAINTING NORMALIZED ERRORS

Regularizer	Pixels	Frames	Total
Tikhonov	0.051	0.100	0.059
TV	**0.048**	0.122	0.060
$\mathrm{N}_{1,1}(\boldsymbol{x})$	0.056	0.059	0.057
$\mathrm{N}_{1,2}(\boldsymbol{x})$	0.050	**0.055**	**0.051**
$\mathrm{N}_{2,1}(\boldsymbol{x})$	0.061	0.103	0.068
$\mathrm{N}_{2,2}(\boldsymbol{x})$	0.053	0.066	0.055

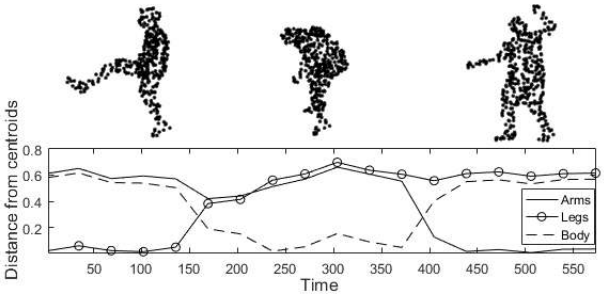

Fig. 10. Clustering of the dancer mesh (no noise): the plot (below) shows the distance of the points stemming from the STVFT representation of three frames (above) of the time-varying mesh closest to the clusters centroids. Each frame shows a different phase of the dance.

C. Overcomplete representations

Last, we examine the utility of STVFT and STVWT, respectively, as a feature extractor for dynamic mesh clustering and as a dictionary used to uncover the wave-like structure and epicenter of a seismic event.

Clustering dynamic meshes using STVFT. We consider the motion classification of a dynamic mesh representing the dancer, corrupted with additive sparse noise with density 0.1 with normally distributed entries and SNR of -20 dB and -10 dB. Our objective is to determine the phase of the dance (*moving arms*, *stretching legs* and *bending body*) at each frames by performing spectral clustering on some representation of the windowed signal. To obtain the ground-truth, we

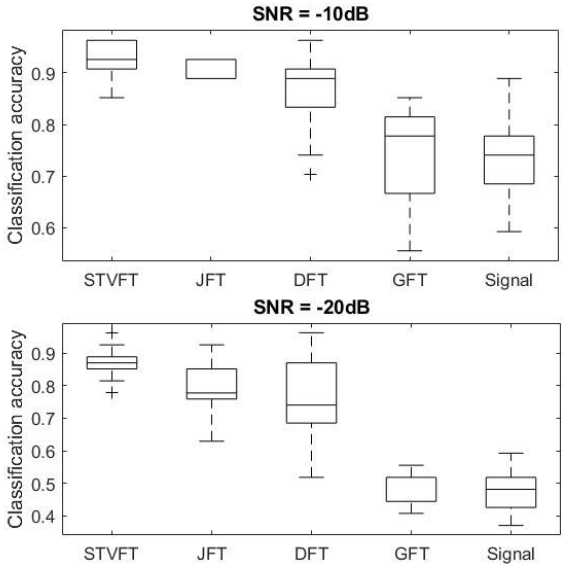

Fig. 11. Comparison of clustering accuracy using different transforms in case of sparse Gaussian noise for SNR -20 dB and -10 dB. Each box shows a summary statistics of the accuracy computed over 20 different realizations of the noise. Results show that SVTFT achieves the highest accuracy in average.

labeled each frame by hand and verified that, when the noiseless signal (i.e., the actual trajectory of the points in time over each window) was used to define the features, one obtains a classification accuracy of 0.926.

Since we want to localize spatial-structured phenomena in time, our approach will be to use a STVFT to derive the representation. To capture the geometry of the problem, we used a nearest neighbour graph constructed based on the coordinates of the mesh vertices averaged in time; this graph was fixed for the whole sequence. As explained in Section V-B, the STVFT is separable, meaning that we can handle the vertex and the time dimensions separately. In the time domain we use a rectangular window with support equal to 50 samples in time and spacing such that the overlap is 60%. For the vertex dimension, we use the an Itersine kernel (defined in the GSPBOX [50]) that we uniformly translate at 5 different positions in the graph spectral domain.

The STVFT provides features associated to a time instant that we can directly use to classify the dance (see Figure 10 for a visual illustration of the clustering results). Other transforms such as GFT, DFT, and JFT do no have this property. Hence, in order to compare with these other transforms, we use the same rectangular windows (width 50, overlap 20 samples) to extract 27 time sequences from the signal. We then used the transformed data associated with each sequence as a point to be clustered.

Figure 11 illustrates the clustering accuracy statistics over 20 realizations of sparse noise for features constructed based on the magnitude of five representations: the windowed sequences, as well as their GFT, DFT, JFT, and STVFT representations. Observe how the presence of sparse noise severely hampers classification when the raw signal is used, with the average accuracy dropping from 0.926 for the clean signals to 0.469 and 0.74 for -20 dB and -10 dB, respectively.

We can also see that the two representations leading to the highest median accuracy are the JFT and STVFT, suggesting the utility of joint harmonic representations. Nevertheless, the STVFT provides more robust estimates with an average accuracy of 0.869 rather than 0.792 for the JFT at -20 dB.

Seismic epicenter estimation with STVWT. We analyze seismic events recorded by the GeoNet sensor network whose epicenters were chosen to be randomly distributed in different areas of New Zealand. We extend the results presented in [21] to a greater dataset using the STVWT with mother kernel based on the wave PDE, which allows us to decompose the signal as sum of PDE solutions. As a first approximation, when the waves propagate in a continuous domain or a regular lattice, seismic waveforms can be modeled as oscillating damped waves [55]. Our premise is thus that we can approximate the seismic waveforms using a small set of damped waves propagating on the graph connecting the seismic stations. Thus, we expect the damped wave mother kernel

$$h(\lambda_\ell, \omega_k) = \frac{1}{\sqrt{T}} \frac{e^{\beta + j\omega_k} + \lambda_\ell/2 - 1}{2(\cosh(\beta + j\omega_k) + \lambda_\ell/2 - 1)} \quad (40)$$

to be a good approximation of the seismic waves recorded by the sensors, with the damping factor β chosen to fit the damping present in the seismic signals. To construct the STVWT, we select 10 equally spaced values in $[0, 2]$ for z_λ and set $z_\omega = 1$. To estimate the epicenter of the earthquake we solve

$$\arg\min_{\boldsymbol{C}} \|D_h^*\{\boldsymbol{C}\} - \boldsymbol{X}\|_2^2 + \gamma \|\boldsymbol{C}\|_1, \quad (41)$$

where γ is the regularization parameter controlling the trade-off between the fidelity term (selected using exhaustive search) and the sparseness assumption and D_h^* is the synthesis operator associated with STVWT.

The solution provides important pieces of information. Firstly, using the synthesis operator we can obtain a denoised version of the original process. Secondly, the non-zero coefficients of $\boldsymbol{C}$, describe the origins and amplitudes of the different components. Therefore, we average the coordinates of the vertices corresponding to the sources of the waves with highest energy coefficients. We compare the performance of STVWT with the estimate obtained using only the signal amplitude: for each earthquake we average the coordinates of the stations using as weights energy of the signals. Figure 12 shows on

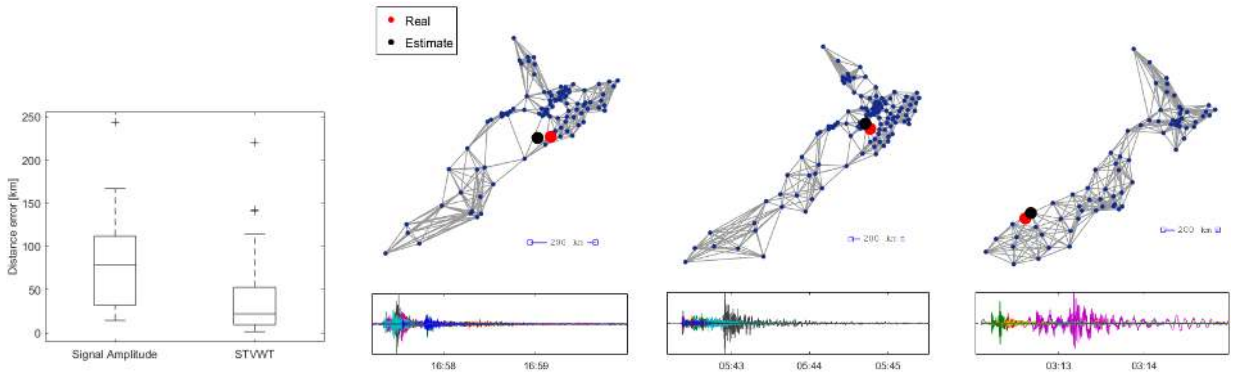

Fig. 12. Left: Comparison of seismic epicenter localization performances between amplitude-based approach and STVWT. The bar graph shows that the second outperforms the first, suggesting that the damped wave model assumption significantly improves the source estimation performance. Right: Results for 3 different seismic events in New Zealand. Right top: the graph is created using the coordinates of the available stations for each event and connecting the closest stations. The stars and the circles are the true and estimated sources of the seismic wave respectively. Right bottom: Signal recorded by the sensors over time for each event.

the left the comparison over 40 different seismic events randomly distributed over the New Zealand between the two methods. STVWT based on the damped wave kernel achieves an average error of 48.5 km, providing an almost twofold improvement over the baseline, whose average performance is 88.3 km. On the right, it illustrates the estimate for 3 different seismic events and the respective seismic waveforms. These results show that the proposed method significantly improves the source estimation performance.

VII. Conclusion

This work puts forth a Time-Vertex Signal Processing Framework, that facilitates the analysis of graph structured data that also evolve in time. We motivate our framework leveraging the notion of partial differential equation on graphs. We introduce joint operators, such as time-vertex localization and we present a novel approach to significantly improve the accuracy of fast joint filtering. We also illustrate how to build time-vertex dictionaries, providing conditions for efficient invertibility and examples of constructions. Our experimental results on a variety of datasets, suggest that the proposed tools can bring forth significant benefits in various signal processing and learning tasks involving time-series on graphs.

Appendix

A. Wave equation

In the continuous setting, the wave equation is

$$\partial_{tt} u - \Delta u = 0$$

where $u \colon \mathbb{R} \times \mathbb{R}^d \to \mathbb{C}$ is a function of both time $t \in \mathbb{R}$ and space $x \in \mathbb{R}^d$, with Δ being the Laplacian operator. Even though being a second order PDE, the equation can be rewritten as a first order system defining $v(t) := \partial_t u(t)$. The pair $(u(t), v(t))$ evolves now according to the system

$$\begin{cases} \partial_t u(t) = v(t) \\ \\ \partial_t v(t) = \Delta u(t) \end{cases} \tag{42}$$

Assuming vanishing initial velocity $v(0) = 0$, the solution $u(t)$ is given via functional calculus by [56]

$$u(t) = \cos(t\sqrt{-\Delta})u(0) \tag{43}$$

where $\cos(t\sqrt{-\Delta})$ is called *propagator operator*.

To obtain a discrete wave equation evolving on a graph, we approximate the second order time derivative with its stencil approximation and the continuous Laplacian Δ with the graph Laplacian $\boldsymbol{L}_G$ with reversed sign:

$$\boldsymbol{X}\boldsymbol{L}_T = s\boldsymbol{L}_G\boldsymbol{X}, \tag{44}$$

where $s > 0$ is the speed of the propagation. The wave equation is a hyperbolic differential equation and several difficulties arise when discretizing it for numerical computation of the solution [56]. Moreover, the graph being an irregular domain, the solution of the above equation is not any more a smooth wave after a few iterations. Nevertheless, we assume as in the case of the heat diffusion Eq. (16) that the solution can be written as

$$\boldsymbol{x}_t = K_s(\boldsymbol{L}_G, t)\boldsymbol{x}_1 = \boldsymbol{K}_{t,s}\boldsymbol{x}_1, \tag{45}$$

where $\boldsymbol{K}_{t,s} = K_s(\boldsymbol{L}_G, t)$ is a matrix obtained applying the function $K_s(\boldsymbol{L}_G, t)$ to the scaled Laplacian $s\boldsymbol{L}_G$ and parametrized by the time t. We will call the operator $\boldsymbol{K}_{t,s}$ "the discrete analogue of the wave propagator" of Eq. (43). Therefore, matrix $\boldsymbol{X} = [\boldsymbol{K}_{t,s}\boldsymbol{x}_1]_{t=1}^{T} = \boldsymbol{K}_s\{\boldsymbol{x}_1\}$ is obtained stacking the

vectors $\boldsymbol{x}_t$ of Eq. (45) along the columns. Substituting (45) into (44), we obtain $\boldsymbol{K}_s\{\boldsymbol{x}_1\}\boldsymbol{L}_T = s\boldsymbol{L}_G\boldsymbol{K}_s\{\boldsymbol{x}_1\}$ which in the graph spectral domain is

$$\widetilde{\boldsymbol{K}}_s\{\tilde{\boldsymbol{x}}_1\}\boldsymbol{L}_T = s\boldsymbol{\Lambda}_G\widetilde{\boldsymbol{K}}_s\{\tilde{\boldsymbol{x}}_1\}, \tag{46}$$

where $\widetilde{\boldsymbol{K}}_{t,s} = K_s(\boldsymbol{\Lambda}_G, t)$. Equation (46) is formally analogous to the eigendecomposition of the operator $\boldsymbol{L}_T$, therefore, the ℓ-th row of $\widetilde{\boldsymbol{K}}_s\{\tilde{\boldsymbol{x}}_1\}$ must be an eigenvector of $\boldsymbol{L}_T$ with eigenvalue λ_ℓ, for every ℓ. It has been proved in [57] that the eigenvectors $\boldsymbol{U}_T(t,k) = \cos(tk\pi/T)$ form also the discrete Fourier basis. Using Eq. (8), we obtain

$$K_s(\lambda_\ell, t) = \cos(t\theta_\ell), \tag{47}$$

with $\theta_\ell = \arccos(1-\frac{s\lambda_\ell}{2})$. Since the $\arccos(x)$ is defined only for $x \in [-1,1]$, to guarantee stability the parameter s must satisfy $s < 4/\lambda_{max}$. We remark that this result is in agreement with the stability analysis of numerical solver for the discrete wave equation presented in [56].

Taking the DFT of the wave kernel in Eq. (47), we obtain

$$\widehat{K}_s(\lambda_\ell, \omega_k) = \sum_t \cos(t\theta_\ell)e^{-j\omega_k t}$$

Therefore, the solution in the joint spectral domain can be written as

$$\widehat{\boldsymbol{X}}(\ell,k) = \widehat{K}_s(\lambda_\ell, \omega_k)\boldsymbol{Z}(\ell,k),$$

where $\boldsymbol{Z}(\ell,k) = \widetilde{\boldsymbol{x}_1}(\ell)\,\boldsymbol{U}_T^*(k,1)$.

Note that there exists a closed form solution for the function $\widehat{K}_s$:

$$\widehat{K}_s(\lambda_\ell, \omega_k) = \begin{cases} \dfrac{\delta(\omega_k+\theta_\ell) + \delta(\omega_k-\theta_\ell)}{2}, & \text{if } T\theta_\ell/2\pi \text{ integer} \\ \dfrac{1}{2}\left(\dfrac{1-e^{-jT(\omega_k+\theta_\ell)}}{1-e^{-j(\omega_k+\theta_\ell)}} + \dfrac{1-e^{-jT(\omega_k-\theta_\ell)}}{1-e^{-j(\omega_k-\theta_\ell)}}\right), & \text{otherwise} \end{cases}$$

B. Frame bound for joint time-vertex dictionaries

Theorem 1. *Let $\{h_z(\lambda,\omega)\}_{z\in\mathcal{Z}_\lambda\times\mathcal{Z}_\omega}$ be the kernels of a time-vertex dictionary $\mathcal{D}_h$, and set*

$$A = \min_{l,k}\sum_z |h_z(\lambda_\ell,\omega_k)|^2,\ B = \max_{l,k}\sum_z |h_z(\lambda_\ell,\omega_k)|^2.$$

If $0 < A \le B < \infty$, then D_h is a frame in the sense:

$$A\|\boldsymbol{X}\|_F^2 \le \|D_h\{\boldsymbol{X}\}\|_F^2 \le B\|\boldsymbol{X}\|_F^2$$

for any time-vertex signal $\boldsymbol{X} \in \mathbb{R}^{N\times T}$.

Proof. In the joint spectral domain we can write:

$$\begin{aligned}
\|\{D_h\{\boldsymbol{X}\}\}\|_2^F &= \sum_{m,\tau,z} |\{D_h\{\boldsymbol{X}\}\}(m,\tau,z)|^2 \\
&= \sum_{z,m,\tau}\left(\sum_{\substack{\ell,k\\n,t}} \boldsymbol{X}(n,t)h_z(\lambda_\ell,\omega_k)\boldsymbol{u}_\ell^*(n)\boldsymbol{u}_\ell(m)e^{-j\omega_k(t-\tau)}\right) \\
&\quad \left(\sum_{\substack{\ell',k'\\n',t'}} \boldsymbol{X}(n',t')h_z(\lambda_{\ell'},\omega_{k'})\boldsymbol{u}_{\ell'}^*(n')\boldsymbol{u}_{\ell'}(m)e^{-j\omega_{k'}(t'-\tau)}\right)^* \\
&= \sum_{z,\ell,k} h_z(\lambda_\ell,\omega_k)\widehat{h}_z^*(\lambda_\ell,\omega_k)\widehat{\boldsymbol{X}}(\ell,k)\widehat{\boldsymbol{X}}^*(\ell,k) \\
&= \sum_{z,\ell,k} |h_z(\lambda_\ell,\omega_k)|^2|\widehat{\boldsymbol{X}}(\ell,k)|^2 = \sum_z \langle |h_z|^2, |\widehat{\boldsymbol{X}}|^2\rangle,
\end{aligned}$$

where the equality holds due to the orthogonality of the eigenvectors. Finally, each element in the sum can be lower bounded and upper bounded by the minimum and maximum value that every filter takes over ℓ and k. Using the Parseval relation (6), the theorem holds. □

REFERENCES

[1] M. De Domenico, A. Lima, P. Mougel, and M. Musolesi, "The anatomy of a scientific rumor," *Scientific reports*, 2013.

[2] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 US election: divided they blog," in *Proceedings of the 3rd international workshop on Link discovery*. ACM, 2005, pp. 36–43.

[3] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," *ACM SIGMOD Record*, no. 2, pp. 17–28, 2013.

[4] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *SENSYS*. ACM, 2008, pp. 323–336.

[5] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, "Epidemic processes in complex networks," *Rev. Mod. Phys.*, no. 3, pp. 925–979, aug 2015.

[6] W. Huang, L. Goldsberry, N. F. Wymbs, S. T. Grafton, D. S. Bassett, and A. Ribeiro, "Graph frequency analysis of brain signals," *IEEE Journal of Selected Topics in Signal Processing*, no. 7, pp. 1189–1203, 2016.

[7] K. Smith, B. Ricaud, N. Shahid, S. Rhodes, J. M. Starr, A. Ibanez, M. A. Parra, J. Escudero, and P. Vandergheynst, "The physiological underpinnings of visual short-term memory binding using graph modular dirichlet energy: Evidence from healthy subjects," *arXiv preprint arXiv:1606.02587*, 2016.

[8] 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," *Signal Process. Mag., IEEE*, no. 3, pp. 83–98, 2013.

[9] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, no. 7, pp. 1644–1656, 2013.

[10] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering." in *NIPS*, 2001, pp. 585–591.

[11] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst, "Fast robust pca on graphs," *IEEE Journal of Selected Topics in Signal Processing*, no. 4, pp. 740–756, 2016.

[12] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *IEEE Trans. Signal Process.*, no. 99, pp. 1–1, 2017.

[13] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *arXiv preprint arXiv:1603.04667*, 2016.

[14] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, no. 2, pp. 129–150, 2011.

[15] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, no. 1, pp. 53–94, 2006.

[16] M. Belkin and P. Niyogi, "Semi-supervised learning on Riemannian manifolds," *Machine learning*, no. 1-3, pp. 209–239, 2004.

[17] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning theory and kernel machines*, B. Schölkopf and M. Warmuth, Eds. Springer, 2003, pp. 144–158.

[18] V. Kalofolias, "How to learn a graph from smooth signals," in *AISTATS*, 2016.

[19] A. Loukas and D. Foucard, "Frequency Analysis of Temporal Graph Signals," in *GLOBALSIP*, 2016.

[20] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Separable autoregressive moving average graph-temporal filters," in *EUSIPCO*. IEEE, 2016, pp. 200–204.

[21] F. Grassi, N. Perraudin, and B. Ricaud, "Tracking time-vertex propagation using dynamic graph wavelets," in *GLOBALSIP*, 2016.

[22] R. Dahlhaus and M. Eichler, "Causality and graphical models in time series analysis," *Oxford Statistical Science Series*, pp. 115–137, 2003.

[23] C. Zhang, D. Florêncio, and P. A. Chou, "Graph Signal Processing–A Probabilistic Framework," Microsoft Research Lab - Redmond, Tech. Rep., 2015.

[24] 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.*, no. 5, pp. 80–90, 2014.

[25] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *Journal of Complex Networks*, no. 3, pp. 203–271, 2014.

[26] K. Benzi, B. Ricaud, and P. Vandergheynst, "Principal patterns on graphs: Discovering coherent structures in datasets," *IEEE Transactions on Signal and Information Processing over Networks*, no. 2, pp. 160–173, 2016.

[27] M. De Domenico, A. Solé-Ribalta, E. Cozzo, M. Kivelä, Y. Moreno, M. A. Porter, S. Gómez, and A. Arenas, "Mathematical formulation of multilayer networks," *Physical Review X*, no. 4, p. 41022, 2013.

[28] A. Loukas, A. Simonetto, and G. Leus, "Distributed autoregressive moving average graph filters," *IEEE Signal Process. Lett.*, no. 11, pp. 1931–1935, 2015.

[29] P. Valdivia, F. Dias, F. Petronetto, C. T. Silva, and L. G. Nonato, "Wavelet-based visualization of time-varying data on graphs," in *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, Oct 2015, pp. 1–8.

[30] J. Mei and J. M. F. Moura, "Signal processing on graphs: Estimating the structure of a graph," in *in ICASSP*. IEEE, 2015, pp. 5495–5499.

[31] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Trans. Signal Process.*, no. 2, pp. 274–288, 2017.

[32] N. Perraudin, A. Loukas, F. Grassi, and P. Vandergheynst, "Towards stationary time-vertex signal processing," in *ICASSP*, 2017.

[33] A. Loukas and N. Perraudin, "Predicting the evolution of stationary graph signals," *arXiv preprint arXiv:1607.03313*, 2016.

[34] A. Loukas and N. Perraudin, "Stationary time-vertex signal processing," *ArXiv e-prints*, Nov. 2016.

[35] D. Burago, S. Ivanov, and Y. Kurylev, "A graph discretization of the laplace-beltrami operator," *Journal of Spectral Theory*, vol. 4, no. 4, pp. 675–714, 2014.

[36] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," *Applied and Computational Harmonic Analysis*, no. 2, pp. 260–291, 2013.

[37] R. Merris, "Laplacian matrices of graphs: a survey," *Linear Algebra and its Applications*, pp. 143 – 176, 1994.

[38] J. Solomon, "PDE approaches to graph analysis," *CoRR*, 2015. [Online]. Available: http://arxiv.org/abs/1505.00185

[39] R. Courant, K. Friedrichs, and H. Lewy, "On the partial difference equations of mathematical physics," *IBM Journal of Research and Development*, no. 2, pp. 215–234, March 1967.

[40] J.-B.-J. Fourier, "Mémoire sur la propagation de la chaleur dans les corps solides," *Nouveau Bulletin des sciences par la Société philomatique de Paris*, p. 112–116, 1807.

[41] T. N. Narasimhan, "Fourier's heat conduction equation: History, influence, and connections," *Reviews of Geophysics*, no. 1, pp. 151–172, 1999. [Online]. Available: http://dx.doi.org/10.1029/1998RG900006

[42] A. Loukas, M. Cattani, M. Zuniga, and J. Gao, "Graph scale-space theory for distributed peak and pit identification," in *IPSN*. ACM/IEEE, 2015.

[43] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *DCOSS*. IEEE, 2011, pp. 1–8.

[44] N. Perraudin, B. Ricaud, D. Shuman, and P. Vandergheynst, "Global and Local Uncertainty Principles for Signals on Graphs," *arXiv preprint arXiv:1603.03030*, 2016.

[45] O. Christensen, *An introduction to frames and Riesz bases (Second Edition)*. Birkhäuser, 2016.

[46] K. Gröchenig, *Foundations of Time-Frequency Analysis*, ser. Applied and Numerical Harmonic Analysis. Boston, MA: Birkhäuser Boston, 2001.

[47] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "A windowed graph fourier transform," in *Statistical Signal Processing Workshop (SSP), 2012 IEEE*. Ieee, 2012, pp. 133–136.

[48] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, "Spectrum-adapted tight graph wavelet and vertex-frequency frames," *IEEE Trans. Signal Process.*, no. 16, pp. 4223–4235, 2015.

[49] J. Kovacevic and A. Chebira, "Life beyond bases: The advent of frames (part i)," *IEEE Signal Process. Mag.*, no. 4, pp. 86–104, July 2007.

[50] N. Perraudin, J. Paratte, D. Shuman, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *ArXiv e-prints*, Aug. 2014.

[51] N. Perraudin, D. Shuman, G. Puy, and P. Vandergheynst, "UNLocBoX A matlab convex optimization toolbox using proximal splitting methods," *ArXiv e-prints*, feb 2014.

[52] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs, *The Large Time-Frequency Analysis Toolbox 2.0*. Cham: Springer International Publishing, 2014, pp. 419–442. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-12976-1_25

[53] A. Buades, B. Coll, and J. M. Morel, "A non-local algorithm for image denoising," in *in CVPR*, June 2005, pp. 60–65.

[54] S. H. Chan, R. Khoshabeh, K. B. Gibson, P. E. Gill, and T. Q. Nguyen, "An augmented lagrangian method for total variation video restoration," *IEEE Trans. Image Process.*, vol. 20, no. 11, pp. 3097–3111, 2011.

[55] W. Lowrie, *Fundamentals of Geophysics*, 2nd ed. Cambridge University Press, 2007.

[56] D. R. Durran, *Numerical methods for wave equations in geophysical fluid dynamics*. Springer Science & Business Media, 2013.

[57] G. Strang, "The discrete cosine transform," *SIAM review*, no. 1, pp. 135–147, 1999.