A Wavelet Lifting Approach for Representing and Denoising Functions on Network Edges

Dingjia Cao
Department of Mathematics, University of York
and
Marina I. Knight
Department of Mathematics, University of York
and
Matthew A. Nunes*
Department of Mathematical Sciences, University of Bath

September 22, 2025

Abstract

Data collected over networks arise in a number of scientific, engineering and industrial applications, in which the datapoints are noisy observations relating to a process of interest over the graph structure. In this article we propose a novel multiscale representation of data on the edges of a network. In contrast to other methods in the literature which employ expensive node to edge data transformations, our decomposition acts directly on the network edges. Using our method, we propose an efficient edge denoising algorithm, termed E-LOCAAT, which displays good performance across a range of data scenarios, particularly when the number of edges is large. The proposed method is illustrated using extensive simulations and we demonstrate its applicability on a real-world dataset arising in road traffic modelling.

Keywords: graphs; multiscale expansion; nonparametric regression; smoothing

*Corresponding author: man54@bath.ac.uk

1 Introduction

In many scientific applications of interest, the primary focus is to understand the dynamics and relationships in systems of interacting entities. These systems can often be represented by network structures, in which edges indicate that interactions exist between the entities, represented by the network nodes. Being able to efficiently model and analyze data collected on these graph structures has the potential to provide insight in diverse fields, from neuroscience (Bullmore and Sporns, 2009), power management (Dörfler et al., 2018), traffic modelling (Kessels, 2019), behaviour in social networks (Doreian and Conti, 2012), financial network analysis (Boginski et al., 2005), as well as environmental processes (Knight et al., 2016).

In the statistical literature, one particular analysis task of interest is to estimate functions on network structures from noisy data. Within this nonparametric regression setting, the majority of techniques focus on function estimation from vertex data, see for example Jansen et al. (2009); Kovac and Smith (2011); Shuman et al. (2013). However, in many scientific areas, where data is observed on the *edges* of the network structure, e.g. chemometrics of river networks (Hoef et al., 2006; O'Donnell et al., 2014; Park and Oh, 2022), traffic flow estimation (Lakhina et al., 2004), or analysis of neural communication data in brain networks (Schwock et al., 2023).

The work in this article is particularly motivated by datasets where noisy observations are obtained on the edges of a network. Figure 1 shows an example dataset of noisy edge observations of traffic flow and associated cost on a road network in Chicago. It is of interest in this context to estimate the underlying function(s) on the graph edges, i.e. denoise the corrupted observations, for example for infrastructure planning, congestion management and maintenance scheduling. In other engineering applications such as telecommunications, accurate smoothing of inter-device packet transmission activity in both Ethernet (Lo Bello et al., 2005) and wireless networks (Buzenkov

et al., 2024) is crucial in server and router load management, for example to impose server traffic limits to reduce the chance of 'packet collisions', known to increase load and degrade network efficiency, see e.g. Kweon and Shin (2003).

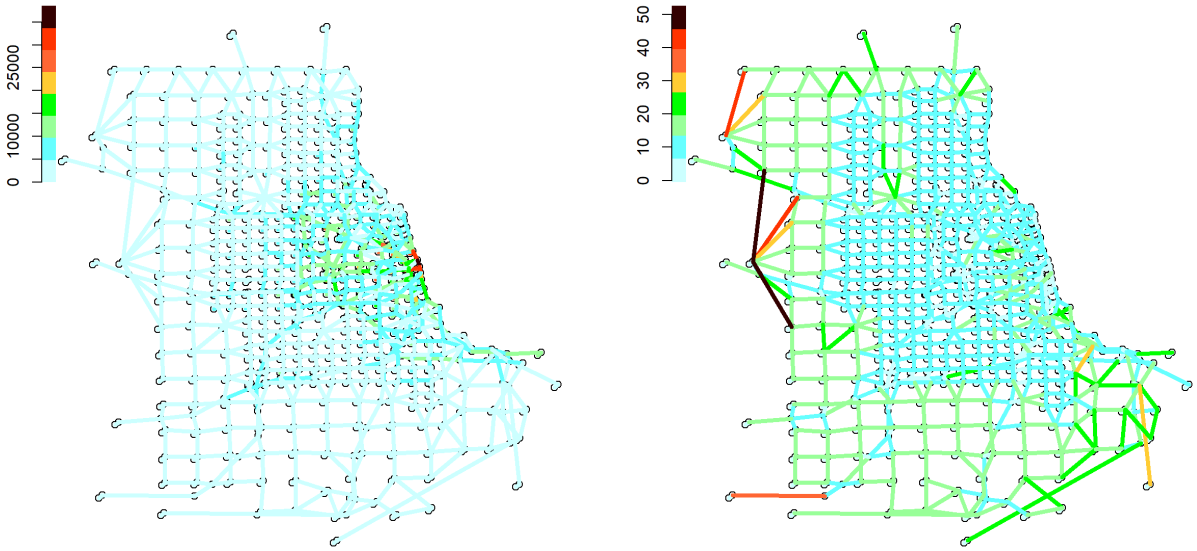

Figure 1: Data observed on the edges of the Chicago-Sketch transportation network. **Left:** Total traffic flow volume. **Right:** Total traffic generalised cost.

For nonparametric regression for data on network edges, available techniques are very limited. In the context of river flow modelling, the methods of O'Donnell et al. (2014) and Park and Oh (2022) use (spline, respectively wavelet lifting) regression techniques on network segments, incorporating the known network structure to form an estimate of the underlying function over the network edge space. Other techniques involve estimation in the vertex space, using pre- and post-processing of noisy edge data to achieve estimation on edges. For example, the recent work of Cao et al. (2024) employ the line-graph transformation together with wavelet lifting techniques to denoise edge signals. Whilst the current state-of-the-art, the method is not guaranteed to be able to represent the data in its original domain at each step, while the line-graph transformation processing also adds undesirable complexity to the algorithm. The 'line-graph denoising' method

described in Schaub and Segarra (2018) also uses the line-graph to form an estimator, and relies on potentially computationally-intensive optimisation procedures to estimate the underlying edge function. All these works are underpinned by the natural assumption that each edge is associated to one value (as opposed to a continuum), an assumption we also adopt for our development.

In this work, we propose a new multiscale technique for directly representing data on network edges, based on the 'lifting one-coefficient-at-a-time' (LOCAAT) transform of Jansen et al. (2009). Through this representation, we are able to denoise functions in the network edge space. Our work builds on the work of Cao et al. (2024), but circumvents the costly line-graph processing step. Our procedure shows improved denoising performance compared to the techniques of Cao et al. (2024) and Schaub and Segarra (2018), as well as demonstrating superior computational efficiency.

This article is structured as follows. Section 2 provides necessary notation and reviews the nonparametric regression problem on graph edges. Section 3 introduces our proposed multiscale network edge-function representation, and incorporates it into an edge-function denoising algorithm. Section 4.1 highlights through simulation the advantages of our approach over current methods. We demonstrate the effectiveness of our technique by applying it to the Chicago traffic volume and cost dataset in Section 4.2.

2 Background

In this section we introduce the pertinent background material that will contribute to our proposed multiscale representation. Section 2.1 gives the necessary graph-theoretic notation that will allow us to define and manipulate functions evaluated at the graph edges, before briefly describing in Section 2.2 the lifting paradigm (its LOCAAT variant appears in Appendix B), a key component of our denoising algorithm that solves the edge-function regression problem of Section 2.3.

2.1 Graphs and metrized graphs

Following the development and notation from Bondy and Murty (1976), we represent a graph G as an ordered pair $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, ..., v_n\}$ is the set of (n) vertices (nodes) and $\mathcal{E} = \{e_1, ..., e_m\}$ is the set of (m) edges. The k-th edge indicates the connection between, say, the i-th and j-th vertices and we write $e_k = \{v_i, v_j\}$, where we assume the edge to be undirected. In what follows, we denote the set of neighbouring vertices of node v_i by $\mathcal{N}_i^{\mathcal{V}} = \{v_j \,|\, v_j \in \mathcal{V};\, \{v_i, v_j\} \in \mathcal{E}\}$.

Weighted graphs provide additional information on the strength of node connection, crucial for our development. Mathematically, a weighted graph G^ω is modelled by an ordered pair (G, ω), where G is the graph topology containing the vertex set and their connectivity (edge set), and $\omega : \mathcal{E} \longrightarrow \mathbb{R}$ is a function that associates a weight to each edge (Bondy and Murty, 1976). As typically assumed in the literature across many application fields (see e.g., Bollobás (1998); Baker and Faber (2006)), we let the weight for each edge $e_k = \{v_i, v_j\} \in \mathcal{E}$ be strictly positive and denote it by ω_k or ω_{ij} interchangeably. The size of the edge weight is intrinsically connected to the particular application, and a large weight on an edge may indicate a weak vertex connection when the weights represent cost, as opposed to a strong connection for e.g., traffic flows. Here, we assume the strength of connection between two vertices v_i and v_j to monotonically increase with the edge weight, namely if v_i and v_j are close to each other, then the weight ω_{ij} tends to be large.

One avenue to facilitate edge function representation is via the *metrized graph* construct. This endows weighted graphs with geometric information and allows us, loosely put, to replace pairs of vertices with line segments. Formally, a metrized graph Γ of a weighted graph G^ω arises from a pair (G, ℓ), where $\ell : \mathcal{E} \longrightarrow \mathbb{R}^+$ is a function that assigns a length $\ell(e) = 1/\omega(e)$ to each edge $e \in \mathcal{E}$ (Baker and Faber, 2006). We associate an interval of length $\ell(e)$ to each edge e, and identify the ends of distinct line segments if they correspond to the same vertex $v \in \mathcal{V}$. The space $\Gamma(G^\omega)$, or

simply Γ, contains all points across these intervals, and G is known as a model for Γ. The distance between two points in Γ is defined as the length of the shortest path between them along the line segments traversed (the path metric, see Appendix A). Endowed with this metric, Γ is a metrized graph on which a vertex and associated edge set may be (non-uniquely) constructed, as follows.

A vertex set ($\mathrm{Vex}(\Gamma)$) is a finite, non-empty subset of Γ containing all points ($\mathbf{p} \in \Gamma$) whose valence is $n_{\mathbf{p}} \neq 2$, where $n_{\mathbf{p}}$ denotes the number of the directions by which a path can leave the point $\mathbf{p}$. The finite set $\Gamma \backslash \mathrm{Vex}(\Gamma)$ is a disjoint union of subspaces $\{U_k\}_k$ isometric to open intervals, where U_k is a neighbourhood of a point $\mathbf{p}_k \in \Gamma$. While the choice of $\mathrm{Vex}(\Gamma)$ is not unique (Baker and Faber, 2006), each vertex set determines a distinct set of metrized edges $\{e_k^{\mathrm{met}}\}_k$, where $e_k^{\mathrm{met}} = \overline{U}_k$ (the topological closure of U_k) such that any distinct metrized edges intersect in at most one point. For the metrized edge e_k^{met} of $e_k = \{v_i, v_j\}$ we also use the alternative notation $[v_i, v_j]$. If Γ is a metrized graph model for graphs G and $\tilde{G}$ and $\mathrm{Vex}_G(\Gamma) \subset \mathrm{Vex}_{\tilde{G}}(\Gamma)$, then $\tilde{G}$ is a refinement of G and we denote $G \sim \tilde{G}$. In general, for two graphs G and G', we denote $G \sim G'$ if they admit a common refinement, namely a graph $\tilde{G}$ satisfying $\mathrm{Vex}_G(\Gamma) \subseteq \mathrm{Vex}_{\tilde{G}}(\Gamma)$ and $\mathrm{Vex}_{G'}(\Gamma) \subseteq \mathrm{Vex}_{\tilde{G}}(\Gamma)$.

2.2 The Lifting Scheme

The lifting scheme is a technique introduced by Sweldens (1996, 1998), which produces wavelet-like biorthogonal bases and associated function representations adapted to general data sampling for which classical wavelets cannot directly work, e.g., functions sampled on complex domains. Since in Section 3 we will introduce a new lifting scheme construction specifically designed to work on the graph *edges*, we next describe the general algorithmic details of a wavelet lifting transform and note that current literature constructions are limited to functions sampled on graph *vertices*.

Considering a sequence of integer-indexed observations on a real line, the lifting scheme consists

of iterating three steps, typically referred to as **split**, **predict** and **update**. **Split** separates the observations into two disjoint sets, one containing the odd-indexed positions and the other the even-indexed ones. **Predict** (in some way) the odd-indexed values using the information from the even-indexed ones, then encode the difference between each observation and its corresponding prediction into the detail, or wavelet, coefficients. **Update** modifies the even-indexed observations using the detail coefficients obtained in the prediction step. Iterating these three steps leads to a set of detail coefficients: in essence, each wavelet coefficient is associated to a wavelet-like basis function, the properties of which are built through the design of the predict and update steps.

2.3 Nonparametric regression over graph edges

In this work we deal with a set of (independent) noisy observations $\{f_k^{\mathcal{E}}\}_{k=1}^m$, where $f_k^{\mathcal{E}}$ is collected from the k-th edge of the graph $G = (\mathcal{V}, \mathcal{E})$ and is represented as

$$f_k^{\mathcal{E}} = g^{\mathcal{E}}(e_k) + \epsilon_k, \tag{1}$$

where $g^{\mathcal{E}} : \mathcal{E} \to \mathbb{R}$ is an underlying (true, unknown) edge-function corrupted by Gaussian noise $\epsilon_k \sim N(0, \sigma^2)$ with zero mean and unknown variance σ^2. Our main goal is to recover the function $g^{\mathcal{E}}$ from the noise contaminated edge observations $\{f_k^{\mathcal{E}}\}_{k=1}^m$. A well-established approach to denoising in the signal processing literature is wavelet thresholding. A wavelet decomposition of the noisy observations is taken, and the resulting wavelet coefficients are thresholded to remove the noise corruption; the wavelet transform is then inverted to recover the true function of interest. For data on irregular sampling domains, LOCAAT algorithms combined with Bayesian thresholding have been shown to perform well for functions with a range of features, see e.g. Nunes et al. (2006).

However, the current state-of-the-art algorithms are only suitable for observations on graph

vertices, and not on edges as in the regression setup (1) we wish to tackle. To adopt a thresholding approach to the edge denoising problem, we need a multiscale algorithm designed to act on (noisy) edge data. We next introduce our proposed edge-based LOCAAT algorithm (E-LOCAAT) and multiscale edge-function representation that facilitate a new network edge denoising technique.

3 Proposed E-LOCAAT constructions

Through the use of the metrized graph concept introduced in Section 2.1, in Section 3.1 we construct wavelet functions generated by interpolating scaling functions within vertex-based and edge-based setups. Their associated graph distance measures are discussed in Section 3.2. These constructions, along with the LOCAAT paradigm (see Appendix B for its brief description), allow us to propose our E-LOCAAT algorithm in Section 3.3, thus facilitating the transform of the original edge-recorded data into (edge-)detail lifting coefficients, used for denoising in Sections 3.4 and 3.5.

3.1 Proposed bases construction

3.1.1 Metrized edge representation

Recall that the unknown true function $g^{\mathcal{E}}$ is defined on the edge set of a graph G for which the length, or the weight, of each edge is available. Accordingly, its associated metrized graph Γ can be obtained and a geometric realisation of the edge-based function can be constructed, where a function $g : \Gamma \to \mathbb{R}$ is defined to be the geometric realisation of $g^{\mathcal{E}}$ if it satisfies

$$g(\mathbf{p}) = g^{\mathcal{E}}(e),\ \forall \mathbf{p} \in e^{\text{met}} \backslash \{\mathbf{p}_v, \mathbf{p}_{v'}\};\ e = \{v, v'\}, \tag{2}$$

$$g(\mathbf{p}) = 0,\ \text{if}\ \mathbf{p} \in \{\mathbf{p}_v, \mathbf{p}_{v'}\}, \tag{3}$$

where the metrized edge e^{met} corresponding to the original edge $e \in \mathcal{E}$ is isometric to a line segment $[0, \ell(e^{\text{met}})]$, with $\ell(e^{\text{met}})$ the length of this edge. Kuchment (2003) uses this construction to define the squared integrable function space on the metrized graph Γ, $L^2(\Gamma)$, which consists of functions that are measurable and integrable on each metrized edge, such that $\|g\|^2_{L^2(\Gamma)} = \sum_{e\in\mathcal{E}} \|g\|^2_{L^2(e^{\text{met}})} < \infty$. The space $L^2(\Gamma)$ can be considered as the orthogonal direct sum of $L^2(e^{\text{met}})$ and as e^{met} is isometric to a closed interval, the inner product can be defined as in $L^2(\mathbb{R})$.

3.1.2 Interpolating-point basis construction and function representation

Denoting the vertex set of the metrized original graph as $\text{Vex}_G(\Gamma) = \mathcal{V}^{\text{met}}$, we propose to build a graph refinement as follows. For the metrized form $e_k^{\text{met}} = [v_i, v_j]$ of an edge $e_k = \{v_i, v_j\}$, we interpolate a point $\mathbf{p}_{ij} \in e_k^{\text{met}}$ (interchangeably denote it as $\mathbf{p}_k$) and then carry out a subdivision by considering the set of points $\{\mathbf{p}_k\}_{k=1}^m$ as a set of interpolating vertices. Thus, we obtain a new metrized vertex set, $\mathcal{V}'^{\text{met}} = \mathcal{V}^{\text{met}} \cup \{\mathbf{p}_k\}_{k=1}^m$ containing $(n+m)$ vertices and for each metrized edge e_k^{met}, we denote $e'^{\text{met}}_{2k-1} = [\mathbf{p}_{v_i}, \mathbf{p}_k]$ and $e'^{\text{met}}_{2k} = [\mathbf{p}_k, \mathbf{p}_{v_j}]$ the new edges resulting from its subdivision.

Since $\mathcal{V}^{\text{met}} \subset \mathcal{V}'^{\text{met}}$, the associated G' is a refinement of G and we propose to construct a set of primal scaling functions $\{\varphi^{\Gamma}_{k,m}\}_{k=1}^m$ on $\Gamma(G')$ that satisfy the following conditions: (i) they are interpolating among the set $\mathcal{V}'^{\text{met}}$, namely $\varphi^{\Gamma}_{k,m}(\mathbf{p}_{k'}) = \delta_{k,k'}$ for $\mathbf{p}_{\mathbf{k}'} \in \mathcal{V}'^{\text{met}}$; (ii) $\overline{\cup_k \text{span}(\varphi^{\Gamma}_{k,m})} = \Gamma$.

Akin to Schröder and Sweldens (1995), as each interpolating-point in the refined graph G' represents an associated edge and an isometry exists between e_k^{met} and the interval $[0, \ell_k]$, we design the scaling function for the k-th edge as a 'triangular' function defined on e_k^{met}. Hence the

k-th point (or vertex)-based initial primal and dual scaling functions are defined as

$$\varphi_{k,m}^{\Gamma,\mathrm{vertex}}(\mathbf{p}) = \begin{cases} 1 - \frac{\mathrm{dist}_{\mathrm{path}}^{\Gamma}(\mathbf{p},\mathbf{p}_k)}{\ell_k/2}, & \text{if } \mathbf{p} \in e_k^{\mathrm{met}}, \\ 0, & \text{if } \mathbf{p} \notin e_k^{\mathrm{met}}; \end{cases} \tag{4}$$

$$\tilde{\varphi}_{k,m}^{\Gamma,\mathrm{vertex}}(\mathbf{p}) = \delta(\mathbf{p} - \mathbf{p}_k), \tag{5}$$

where $\delta(\cdot)$ is the Dirac delta, and '$\mathrm{dist}_{\mathrm{path}}^{\Gamma}(\mathbf{p}, \mathbf{p}_k)$' is the path distance between $\mathbf{p}$ and $\mathbf{p}_k$ on the metrized graph Γ, as described in Appendix A. The interpolating property of the primal scaling functions and their biorthogonality with the dual scaling functions can be readily shown, such that $\langle \tilde{\varphi}_{k,m}^{\Gamma,\mathrm{vertex}}, \varphi_{k',m}^{\Gamma,\mathrm{vertex}} \rangle = \delta_{k,k'}$, where as above $\delta_{k,k'}$ denotes the Kronecker delta.

Function representation. Since we can represent the edge-defined function $g^{\mathcal{E}}$ as g^{Γ} by means of the metrized graph, where $g_k^{\mathcal{E}} := g^{\Gamma}(\mathbf{p}_k)(= g_k^{\Gamma})$, for $k \in \{1, ..., m\}$ and using $\langle \tilde{\varphi}_{k,m}^{\Gamma,\mathrm{vertex}}, g^{\Gamma} \rangle = g_k^{\Gamma}$, we obtain the (initial) function representation

$$g^{\mathcal{E}}(e) = g^{\Gamma}(\mathbf{p}) = \sum_{k=1}^{m} c_{k,m}^{\Gamma} \varphi_{k,m}^{\Gamma,\mathrm{vertex}}(\mathbf{p}), \tag{6}$$

where $\mathbf{p} \in \Gamma$, and the initial scaling coefficients are $c_{k,m}^{\Gamma} := \langle \tilde{\varphi}_{k,m}^{\Gamma,\mathrm{vertex}}, g^{\Gamma} \rangle = g_k^{\Gamma}$.

An alternative proposal for the initial primal and dual scaling functions, that still allows for a representation akin to (6), follows in the same vein as the 'lazy wavelet' construction (Sweldens, 1998), which in this context amounts to taking the biorthogonal pairs

$$\varphi_{k,m}^{\Gamma,\mathrm{Delta}}(\mathbf{p}) = \delta_{\mathbf{p}_k,\mathbf{p}}; \tag{7}$$

$$\tilde{\varphi}_{k,m}^{\Gamma,\mathrm{Delta}}(\mathbf{p}) = \delta(\mathbf{p} - \mathbf{p}_k). \tag{8}$$

3.1.3 Edge basis construction and function representation

We now introduce a construction of edge bases, as a variant of the face bases of Schröder and Sweldens (1995). We first aim to construct a suitable partitioning for the metrized graph domain Γ, and then ensure the domain of each scaling function lies in one of the partitionings (Jawerth and Sweldens, 1994). One such natural partitioning is obtained from the metrized edge construction as the set $\{\mathbf{P}_k\}_{k=1}^m$, where for the k-th edge $e_k = \{v_i, v_j\}$, we let $e_k^{\text{met}} = [\mathbf{p}_{v_i}, \mathbf{p}_{v_j}]$ be its metrized version and denote $\mathbf{P}_k = \text{int}(e_k^{\text{met}})$ as its interior, namely $e_k^{\text{met}} \backslash \{\mathbf{p}_{v_i}, \mathbf{p}_{v_j}\}$. The isometry between e_k^{met} and $[0, \ell_k]$ induces an isometry between $\mathbf{P}_k$ and the open interval $(0, \ell_k)$, leading to the proposal to construct the initial edge-based primal and dual scaling functions as

$$\varphi_{k,m}^{\Gamma,\text{edge}}(\mathbf{p}) = \chi_{\mathbf{P}_k}(\mathbf{p}) := \chi_{(0,\ell_k)}, \tag{9}$$

$$\tilde{\varphi}_{k,m}^{\Gamma,\text{edge}}(\mathbf{p}) = \frac{1}{\mu(\mathbf{P}_k)} \chi_{\mathbf{P}_k}(\mathbf{p}) := \frac{1}{\ell_k} \chi_{(0,\ell_k)}, \tag{10}$$

where χ is the characteristic function, and $\mu(\mathbf{P}_k)$ is the Lebesgue measure of the interval $(0, \ell_k)$, which is simply the length of its closure. The interpolating property of the primal scaling functions and their biorthogonality with the corresponding duals follow, as shown in Appendix C. Hence the initial coefficients are $c_{k,m}^{\Gamma} := g_k^{\Gamma}$ and the stage-m functional expansion can be written as in (6).

We also investigate a variant of the edge basis framework, which we shall refer to as 'biorthogonal Haar' basis, a term coined by Schröder and Sweldens (1995). Their construction is detailed in Appendix D and guarantees the self-similarity of the scaling and (associated) wavelet functions, with the wavelet functions taking the form of Haar-like step functions on the metrized edges.

Remark. The appeal of the proposals in this section over the LG-LOCAAT algorithm of Cao et al. (2024) is that the scaling functions are defined *directly* on the metrized form (Γ) of the original

graph (G) instead of having to resort to its line-graph. We will explore this further in Section 4.1.2.

3.2 Distance measures

When compared with interpolating-point bases, edge bases do not use points as an edge representation, but treat each edge as a set of points instead. Hence the interpolating-point bases and edge bases will give rise to different distance measures on the graph. This in turn will play an important role in the proposed lifting algorithm, and in particular for the construction of prediction weights.

Distance for interpolating-point bases. Denoting by Γ_r the metrized version for the graph at stage-r, we define the stage-r distance between two neighbouring edges e_{k_r} and e_s by means of the path distance between their metrized versions,

$$\mathrm{dist}_{\mathrm{path}}^{\Gamma_r,r}(\mathbf{p}_{k_r}, \mathbf{p}_s) = \frac{\ell_{k_r,r} + \ell_{s,r}}{2}. \tag{11}$$

Distance for edge bases. A suitable metric that gives the distance between two sets of points is the Hausdorff distance (see O'Searcoid (2006) for its definition) which, along with some of its variants, is widely used in image analysis, e.g. Huttenlocher et al. (1993); Zhao et al. (2005); Karimi and Salcudean (2019). Using the path metric in equation (11) and the metric space $(\Gamma_r, \mathrm{dist}_{\mathrm{path}}^{\Gamma_r,r})$ at stage-r, if $e_s = \{v_j, v_l\}$ is neighbouring $e_{k_r} = \{v_i, v_j\}$, their Hausdorff distance follows as

$$\begin{aligned}
\mathrm{dist}_H(e_{k_r}^{\mathrm{met}}, e_s^{\mathrm{met}}) &= \max\left\{ \sup_{\mathbf{p}\in e_{k_r}^{\mathrm{met}}} \left\{ \inf_{\mathbf{p}'\in e_s^{\mathrm{met}}} \left\{ \mathrm{dist}_{\mathrm{path}}^{\Gamma_r,r}(\mathbf{p},\mathbf{p}') \right\} \right\}, \sup_{\mathbf{p}'\in e_s^{\mathrm{met}}} \left\{ \inf_{\mathbf{p}\in e_{k_r}^{\mathrm{met}}} \left\{ \mathrm{dist}_{\mathrm{path}}^{\Gamma_r,r}(\mathbf{p},\mathbf{p}') \right\} \right\} \right\} \\
&= \max\left\{ \sup_{\mathbf{p}\in e_{k_r}^{\mathrm{met}}} \left\{ \mathrm{dist}_{\mathrm{path}}^{\Gamma_r,r}(\mathbf{p},\mathbf{p}_{v_j}) \right\}, \sup_{\mathbf{p}'\in e_s^{\mathrm{met}}} \left\{ \mathrm{dist}_{\mathrm{path}}^{\Gamma_r,r}(\mathbf{p}',\mathbf{p}_{v_j}) \right\} \right\} = \max\left\{ \ell_{k_r,r},\, \ell_{s,r} \right\}.
\end{aligned}$$

3.3 Proposed E-LOCAAT algorithm

Our proposed algorithm designed to directly work on the network edges will be formalised by the iterative split-predict-update paradigm (Section 2.2 and Appendix B), as detailed next. While Section 3.1 described the construction of the initial scaling functions by two different approaches (interpolating-point/edge bases), there are only a few instances where these choices impact the proposed algorithm, hence we will specifically point out the differences where these occur.

Split step. The first task for the 'split' step is to determine the integral values of the primal scaling functions at the initial stage, here stage-m. We propose to remove the edge corresponding to the minimum integral, ensuring we explore the denser sampled regions of the network first. Should several edges correspond to the same value, we choose at random an edge for removal from this set. Denote the edge chosen for removal by e_{k_r} for a general stage-r with $r = m,\, m-1, \ldots,\, 3$.

The integral derivations for the k-th edge scaling function are detailed in Appendix E and summarised next: (i) for the interpolating-point bases in (4), we have $I^{\Gamma,\text{vertex}}_{k,m} = \ell_{k,m}/2$; (ii) for the Kronecker delta as the initial primal scaling functions in (7), we have $I^{\Gamma,\text{Delta}}_{k,m} = 1$; and finally, (iii) for the edge bases construction in (9), we obtain $I^{\Gamma,\text{edge}}_{k,m} = \ell_{k,m}$, where we recall that $\ell_{k,m}$ denotes the initial (stage-m) length of k-th edge. As the scaling function choice will not impact the scaling and wavelet coefficients, below we skip the superscript (vertex/Delta/edge) for brevity.

Predict step. The interpolating property of the primal scaling functions ensures that each $e_k \in \mathcal{E}$ is associated to an initial scaling coefficient value $c^{\Gamma}_{k,m} := g^{\Gamma}_k$ on the metrized graph domain Γ. To ensure full generality, we present the stage-r algorithm, where the predicted value for the removal edge, e_{k_r}, is obtained by a linear combination of the neighbouring coefficients $\{c^{\Gamma}_{s,r}\}_{s:e_s\in\mathcal{N}^{\mathcal{E}}_{k_r,r}}$, where the set $\mathcal{N}^{\mathcal{E}}_{k_r,r}$ encompasses the edges that have a common vertex with the edge e_{k_r} at stage-r.

Akin to (B.3), the detail (or wavelet) coefficient d_{k_r} encapsulates the residual from prediction

$$d^{\Gamma}_{k_r} = c^{\Gamma}_{k_r,r} - \sum_{s:\, e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} a^{\Gamma}_{s,r} c^{\Gamma}_{s,r}, \tag{12}$$

where $\{a^{\Gamma}_{s,r}\}_s$ are the prediction weights satisfying $\sum_{s:\, e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} a^{\Gamma}_{s,r} = 1$. For brevity, when the clarity is not affected, in the remainder of the paper we drop the superscript Γ for all related quantities.

Since the interpolating-point bases and edge bases gave rise to different distance measures (Section 3.2), the corresponding prediction weights (see also equation (B.4)) are given by

$$a^{\text{vertex}}_{s,r} = \frac{1/(\ell_{k_r,r} + \ell_{s,r})}{\sum_{j:\, e_j \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} 1/(\ell_{k_r,r} + \ell_{j,r})}, \text{ for } s:\, e_j \in \mathcal{N}^{\mathcal{E}}_{k_r,r}, \tag{13}$$

$$a^{\text{edge}}_{s,r} = \frac{1/\max\{\ell_{k_r,r},\, \ell_{s,r}\}}{\sum_{j:\, e_j \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} 1/\max\{\ell_{k_r,r},\, \ell_{j,r}\}}, \text{ for } s:\, e_j \in \mathcal{N}^{\mathcal{E}}_{k_r,r}. \tag{14}$$

The superscripts 'vertex' and 'edge' indicate the correspondence to the interpolating-point bases and edge bases, respectively. Due to the minimal integral (thus, minimal length) condition for edge removal, the prediction weights for edge bases in (14) can simply be further represented as

$$a^{\text{edge}}_{s,r} = \frac{1/\ell_{s,r}}{\sum_{j:\, e_j \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} 1/\ell_{j,r}}, \text{ for } s:\, e_j \in \mathcal{N}^{\mathcal{E}}_{k_r,r}. \tag{15}$$

The weight construction above is also employed with a 'Delta' choice split step.

Update step. Three quantities are updated at each stage, namely the scaling function integral and coefficient values (equivalent to (B.5)– (B.6)), and the graph structure is relinked. Our proposed E-LOCAAT will additionally require the lengths of the neighbouring edges to be updated.

The integral and coefficient values of the neighbouring edges of e_{k_r} are updated at stage-$(r-1)$

$$I_{s,r-1} = I_{s,r} + a_{s,r} I_{k_r,r}, \text{ for } s : e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r},$$
$$c_{s,r-1} = c_{s,r} + b_{s,r} d_{k_r}, \text{ for } s : e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}. \tag{16}$$

Since the prediction weights are positive quantities, the integral values of the scaling functions will be non-decreasing as the algorithm progresses. The values of the update coefficients can be obtained by taking the minimum norm solution of the underdetermined system $\sum_{s:e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} b_{s,r} I_{s,r-1} = I_{k_r,r}$. For E-LOCAAT, updating (relinking) the graph structure occurs much more naturally than by using the minimal spanning tree as for the vertex-based LOCAAT. We propose to equate the removal of an edge $e_{k_r} = \{v_{i_r}, v_{j_r}\}$ with its two vertices fusing (Figure 2), crucially ensuring that the underlying graph can be obtained at each step of the algorithm. The updated lengths are

$$\ell_{s,r-1} = \ell_{s,r} + \frac{1}{2}\ell_{k_r,r}, \text{ for } s : e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}, \text{ or} \tag{17}$$
$$\ell_{s,r-1} = \ell_{s,r} + a_{s,r}\ell_{k_r,r}, \text{ for } s : e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}, \tag{18}$$

when we notionally consider the two vertices to fuse at the middle point of the associated metrized edge, or use the same scheme as for the integral value update (leading to a lower computational effort). We refer to these as 'unweighted' (17) and 'weighted' (18) length updates, preserving average edge length and ensuring that the lengths are non-decreasing from stage-r to stage-$(r-1)$.

Iterate. We reiterate the split-predict-update steps and obtain the detail coefficients, $\{d_{k_m}, ..., d_{k_{\tau+1}}\}$, where $\tau \in \mathbb{Z}$ is the number of retained edges (or stopping time). The resulting detail coefficients are defined directly on the edge topology, thus avoiding additional constructions necessary to ensure the applicability of existing vertex-based approaches (LG-LOCAAT of Cao et al. (2024)) and,

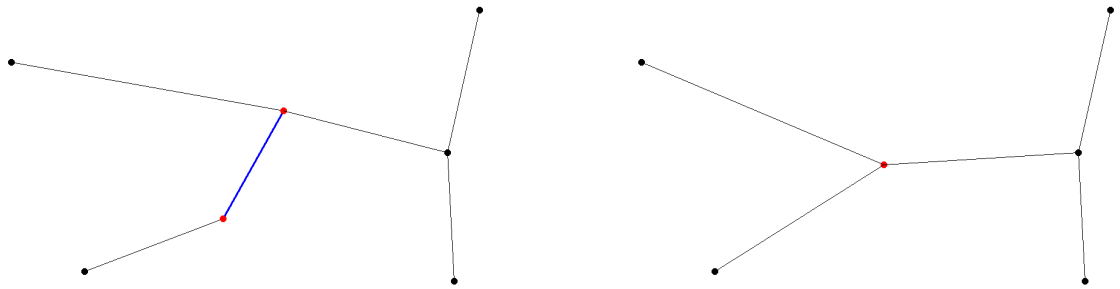

Figure 2: Edge removal visualisation: original graph (red) vertices (left) fuse into a new one (right).

importantly, allowing for a practical and interpretable multiscale representation of the network.

Inversion. The inverse E-LOCAAT transform can be carried out by undoing at each stage equations (12) and (16), as follows

$$c_{s,r} = c_{s,r-1} - b_{s,r} d_{k_r}, \text{ for } s \text{ such that } e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r},$$
$$c_{k_r,r} = d_{k_r} + \sum_{s:e_s \in \mathcal{N}^{\mathcal{E}}_{k_r,r}} a_{s,r} c_{s,r}.$$

Algorithmic comparison of proposed E-LOCAAT with existing line-graph lifting constructions (Cao et al., 2024). For a graph with n nodes and m edges, recall its line-graph has m vertices and $\sum_{i=1}^{n} \binom{n_i}{2} = \frac{1}{2}\left(\sum_i n_i^2\right) - m$ edges, where n_i denotes the number of neighbours (degree) of vertex v_i (see e.g. Harary (2018, Theorem 8.1)). For a dense graph with a small, but highly connected, number of vertices (resulting in large m and n_i values), working with its corresponding line-graph has the undesirable result of having to deal with a large number of vertices and edges.

Importantly, the (average) degree of nodes in the line-graph of a network will increase as a result of the transformation, scaling from $\frac{1}{n}\sum_{i=1}^{n} n_i$ in the original graph to $\frac{1}{m}\sum_{i=1}^{n} n_i^2 - 2$ in the line-graph, which will hold for all graphs, regardless of the generating process (including the

minimal spanning tree constructions implemented in this article). This mirrors the observation for common graph structures such as Erdös-Rényi (Wang et al., 2016), exponential and scale-free models (Nacher et al., 2005; Mańka-Krasoń et al., 2010) that the line-graph produces networks with higher degrees and more clustered edges, i.e larger cliques.

The predict step (specified by equations (B.3) and (B.4)), as well as the update step (equations (B.5)–(B.7)) for the LG-LOCAAT approach will thus require additional computations compared to the corresponding lifting steps of the proposed E-LOCAAT algorithms, see also Section 4.1.2.

3.4 Denoising strategy using the E-LOCAAT representation

We next use our proposed multiscale representations in the context of the network edge nonparametric regression problem described in Section 2.3. Recall we model the edge observations as $f_k^{\mathcal{E}} = g^{\mathcal{E}}(e_k) + \epsilon_k \, (:= g_k^{\mathcal{E}} + \epsilon_k)$, where $g^{\mathcal{E}}$ is the true, unknown function defined on the network edge set $\mathcal{E} = \{e_k\}_{k=1}^m$, and $\{\epsilon_k\}_{k=1}^m$ is iid noise, assumed to follow a normal distribution $N(0, \sigma^2)$. Using the linearity of the E-LOCAAT transform and denoting its associated transform matrix by $\tilde{W}$, we re-write the regression above as $\tilde{W}\underline{f}^{\mathcal{E}} = \tilde{W}\underline{g}^{\mathcal{E}} + \tilde{W}\underline{\epsilon}$, where $\underline{f}^{\mathcal{E}}$, $\underline{g}^{\mathcal{E}}$, and $\underline{\epsilon}$ are the vector forms of the observations, true values, and noise, respectively. As $\underline{d}^{\star} := \tilde{W}\underline{g}^{\mathcal{E}}$ is a sparse vector, while $\tilde{W}\underline{\epsilon}$ is a vector of heteroscedastic Gaussian random noise, a possible solution for obtaining an estimator of $g^{\mathcal{E}}$ is to perform wavelet thresholding of the observed wavelet coefficients $\underline{d} := \tilde{W}\underline{f}^{\mathcal{E}}$.

Denoting by $V_{k_r}^{\mathcal{E},(d)}$ the variance of the detail coefficient and by $V_{s,r}^{\mathcal{E}}$ the variance of the s-th scaling coefficient at stage-r, we incorporate a normalisation step (Jansen et al., 2009) that uses

$$V_{k_r}^{\mathcal{E},(d)} = V_{k_r,r}^{\mathcal{E}} + \sum_{s:\, e_s \in \mathcal{N}_{k_r,r}^{\mathcal{E}}} (a_{s,r})^2 \, V_{s,r}^{\mathcal{E}},$$

$$V_{s,r-1}^{\mathcal{E}} = (1 - 2a_{s,r}b_{s,r}) \, V_{s,r}^{\mathcal{E}} + (b_{s,r})^2 \, V_{k_r}^{\mathcal{E},(d)}.$$

We let $\mathbf{nml}\left(d_{k_r}\right) = d_{k_r} \big/ \sqrt{V_{k_r}^{\mathcal{E},(d)}}$ be the normalised k_r-th detail coefficient, for all $r \in \{m, ..., 3\}$, satisfying $\mathrm{Var}\left(\mathbf{nml}\left(d_{k_r}\right)\right) \approx \sigma^2$, with the inaccuracy stemming from ignoring the induced covariance structure, shown to only cause a small loss of precision (Nunes et al., 2006). Empirical Bayes thresholding is carried out, where the non-zero part of the prior density is modelled as the 'quasi-Cauchy' distribution from Johnstone and Silverman (2004), also employed in other LOCAAT-based approaches (Jansen et al., 2009; Cao et al., 2024). Following thresholding, we obtain the estimated wavelet coefficients $\underline{\hat{d}}^{\star}$, and then the inverse E-LOCAAT transform will yield the estimates, $\{\hat{g}_k^{\mathcal{E}}\}_{k=1}^m$.

3.5 Nondecimated E-LOCAAT variants

For an equitable comparison to the results shown in Park and Oh (2022) and Cao et al. (2024), we bring the nondecimation concept into the denoising E-LOCAAT algorithms introduced in this paper. Knight and Nason (2009) proposed a nondecimated lifting transform (NLT) which saw applications in denoising (Nunes et al., 2006), spectral estimation (Knight et al., 2012) and long-memory estimation (Knight et al., 2017; Knight and Nunes, 2019). Instead of defining shift operators as in other wavelet transforms (Nason and Silverman, 1995), different 'trajectories' of removal order are used for exploring the whole space. In the context here, nondecimation works as different trajectories give rise to different estimates of $g^{\mathcal{E}}$. An overall estimate can then be formed by averaging (mean/ median) the estimates from P (randomly sampled) trajectories, akin to ensemble learning.

We will also test a completely random biorthogonal Haar NLT algorithm (see Appendix F), in which *both* the removal edge and the neighbour used for prediction are chosen randomly. For all other E-LOCAAT algorithms, their nondecimated variant is obtained using random trajectories. In the reported results, these algorithms are indicated by the acronym 'nlt'.

Further potential research avenues are to investigate whether the proposed method can be

extended to higher order prediction steps, and therefore adapted to local smoothness conditions.

4 Data Analysis

4.1 Simulation study

In this section we report the headline findings of our investigations into the behaviour of our proposed E-LOCAAT in the network edge estimation problem of Section 3.4. A summary of the acronyms corresponding to different E-LOCAAT variants appears in Table 1. The full details and results are provided in Appendix G.

Acronyms	
E-Lid-wu	L: lengths as initial integral values (equation (E.15)); id: inverse distance prediction (equation (13)); wu: weighted update for edge lengths (equation (18))
E-Lid-nwu	L: lengths as initial integral values (equation (E.15)); id: inverse distance prediction (equation (13)); nwu: equally-weighted update for edge lengths (equation (17))
E-Lil-wu	L: lengths as initial integral values (equation (E.15)); il: inverse length prediction (equation (15)); wu: weighted update for edge lengths (equation (18))
E-Lil-nwu	L: lengths as initial integral values (equation (E.15)); il: inverse length prediction (equation (15)); nwu: equally-weighted update for edge lengths (equation (17))
E-Did-wu	D: sequence of ones as initial integral values; id: inverse distance prediction (equation (13)); wu: weighted update for edge lengths (equation (18))
E-Did-nwu	D: sequence of ones as initial integral values; id: inverse distance prediction (equation (13)); nwu: equally-weighted update for edge lengths (equation (17))
E-Dil-wu	D: sequence of ones as initial integral values; il: inverse length prediction (equation (15)); wu: weighted update for edge lengths (equation (18))
E-Dil-nwu	D: sequence of ones as initial integral values; il: inverse length prediction (equation (15)); nwu: equally-weighted update for edge lengths (equation (17))
Bio-Haar (BH)	Biorthogonal Haar E-LOCAAT (Appendix D)

Table 1: Acronyms and algorithm descriptions for different parameter choices of E-LOCAAT.

Simulation setup, competitor methods and performance metrics. We simulate noisy datasets in a range of scenarios, and test our algorithms' ability to recover the true uncorrupted edge signals. The simulation study assesses the denoising performance of our proposed E-LOCAAT algorithms against state-of-the-art edge denoising methods in the literature. Specifically, our simulation study compares E-LOCAAT to the recent LG-LOCAAT proposal of Cao et al. (2024) which tackles estimation via a LOCAAT application on the line graph transform of the data. We also include the 'line-graph denoising' method of Schaub and Segarra (2018), which we denote as 'SS'. This technique estimates the underlying edge function via $\tilde{\underline{g}}^{\mathcal{E}} = (I + \alpha L_{LG})^{-1}\underline{f}^{\mathcal{E}}$, where I is the $(m \times m)$ identity matrix, L_{LG} denotes the line-graph Laplacian, and α is a tuning parameter, which in practice is determined via grid search (we use a grid of 50 equally-spaced values for α in $(0, 50]$). To ensure benchmark comparisons, we sample the pointwise and edge averaging constructed functions investigated in Cao et al. (2024) over $q = 1, ..., Q$ different graph structures, $G^{(q)}$. For each graph we simulate $t = 1, ..., T$ different noise sequences, each corresponding to a particular signal-to-noise ratio (SNR). Following the estimation procedure detailed in Section 3.4, we obtain the estimate $\hat{g}^{\mathcal{E}}_{k,q,t}$ of the true k-th edge function value ($g^{\mathcal{E}}_{k,q}$) on the graph $G^{(q)}$ when the true function is corrupted by the t-th noise sequence and report the AMSE and squared bias associated with our methods (see Appendices G and H for definitions and comprehensive results).

4.1.1 Denoising results

Tree structures. For both pointwise and edge averaging function constructions, results in Tables G.3 and G.8 for tree structures with $m = 99$ edges show that the algorithm performance is consistent across different SNR levels. Similar to the results obtained by the LG-LOCAAT algorithm in Cao et al. (2024), the denoising performance is highly related to the type of underlying function. For smooth functions, E-LOCAAT is competitive compared with LG-LOCAAT (and

compares even more favourably for edge averaging functions), but less competitive for functions with discontinuities ('Blocks', g_1, and `mfc`), except for the biorthogonal Haar E-LOCAAT which significantly surpasses every other method for g_1 and the Blocks function. As for LG-LOCAAT, E-LOCAAT also does not perform well for high frequency functions, such as Heavisine. We note that the SS method performs particularly well for the Heavisine function, where the lifting-based methods perform worst. Investigating the effect of scaling the number of edges to denoise for larger tree structures (with $m = 299$, 499 edges) reveals broadly similar conclusions for both pointwise and edge averaging functions (Tables G.4, G.9 and Tables G.5, G.10 respectively), but with the lifting-based methods performing better apart from the Heavisine function. Overall, the 'Did-nwu' method shows more performance consistency, with improvements over other variants regardless of the data generating process, while for the line-graph LG-LOCAAT algorithm, the best performing method switches from 'Aid-p' to 'Did-p' when moving from pointwise to edge averaging.

Denser graphs. To further (comparatively) investigate the denoising performance on different (non-tree) graph structures, we also explored simulated test data on denser graph structures. We constructed a graph with $n = 40$ nodes with m edges corresponding to the m shortest Euclidean distances between the nodes. This will naturally form clusters of edges or 'cliques' in the graph structure. Tables G.6 and G.7 for graphs with two different densities ($m = 299$, 499 edges) highlight that all methods worsen in performance compared to the tree graph structures. However, the lifting-based methods, particularly the biorthogonal Haar construction, are the best in many scenarios. Inspecting the bias metrics for these graphs in Table H.17 there is a clear bias-variance trade-off, with a low estimator variance achieved at the expense of a very large bias.

4.1.2 Comparison of computational cost of denoising methods

Since the network structures encountered in many applications may feature a large number of edges, it is of practical importance to evaluate the feasibility of the methods via their computational cost. To this end, we run each denoising method on a set of simulated data on graphs with increasing number of edges and record its runtime, averaged over $Q = 50$ graph structures and $T = 100$ noise replications. This allows us to compare how the runtime of each method scales with the number of edges, removing the effect of other factors such as implementation efficiency and programming language. The results of the runtime analysis appear in Figure 3. As expected, the relative computational cost of all methods increase with the number of edges in the graphs under analysis.

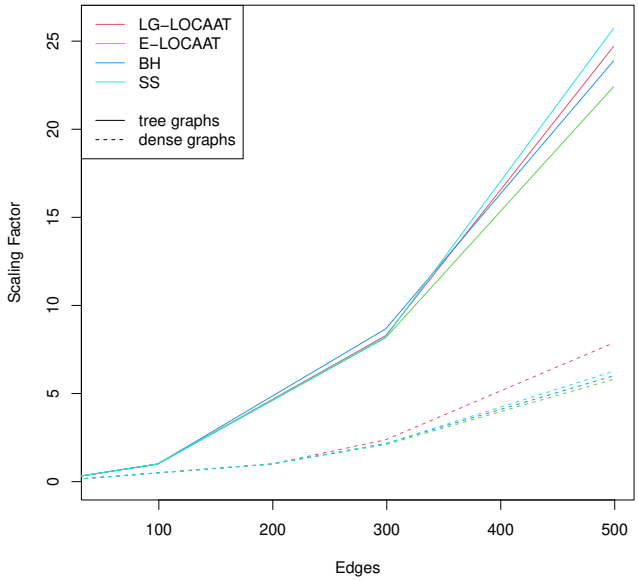

Figure 3: Comparison of computational cost of the various denoising methods in terms of how the computational burden scales with the number of edges in graph structures analysed.

Interestingly, all methods have worse runtime behaviour for tree-like graph structures when compared to the denser graph construction. However, it is noteworthy that the E-LOCAAT and biorthogonal Haar (BH) methods proposed in this article scale better with the number of edges in the graph, when compared to both the LG-LOCAAT and SS competitor methods. Similar scaling

behaviour as Figure 3 was observed regardless of the test function or signal-to-noise ratio used to generate the simulated datasets.

This observation and the good denoising performance over the range of data generation scenarios strongly justify the use of the proposed methods, especially our biorthogonal Haar construction.

4.1.3 Denoising comparisons on the simulated flow-based function

We further compare E-LOCAAT with the methods of O'Donnell et al. (2014); Park and Oh (2022) on their simulated flow function (Figure 4) over a river network with 81 vertices and 80 edges, its edge set being separated into 7 different clusters. (See Appendix I for a detailed analysis.)

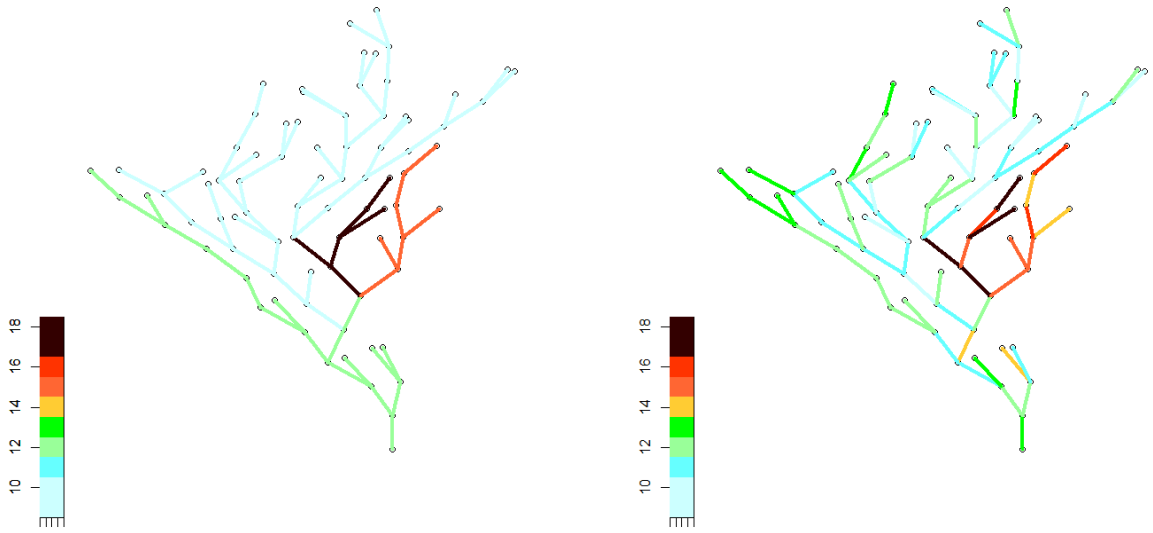

Figure 4: **Left:** The 'true' simulated river flow data used in Cao et al. (2024), the network structure is originally introduced in Gallacher et al. (2017), and the test function construction intruduced by Park and Oh (2022). **Right:** The flow data corrupted by noise $\epsilon \sim N(0, \sigma^2)$, where $\sigma = 1.5$.

Results. We tested 'E-Did-nwu' and 'E-Lil-nwu' as the former was shown to yield good results for test functions which display similar characteristics to the function here, and the latter gives an algorithm with a different focus, treating each edge as a set of points (see Section 3.1.3); biorthogonal Haar ('Bio-Haar', or BH) is also tested since it works well for piecewise constant functions. Results in Table I.23 show that the one trajectory line-graph 'LG-Sid-p' results surpass

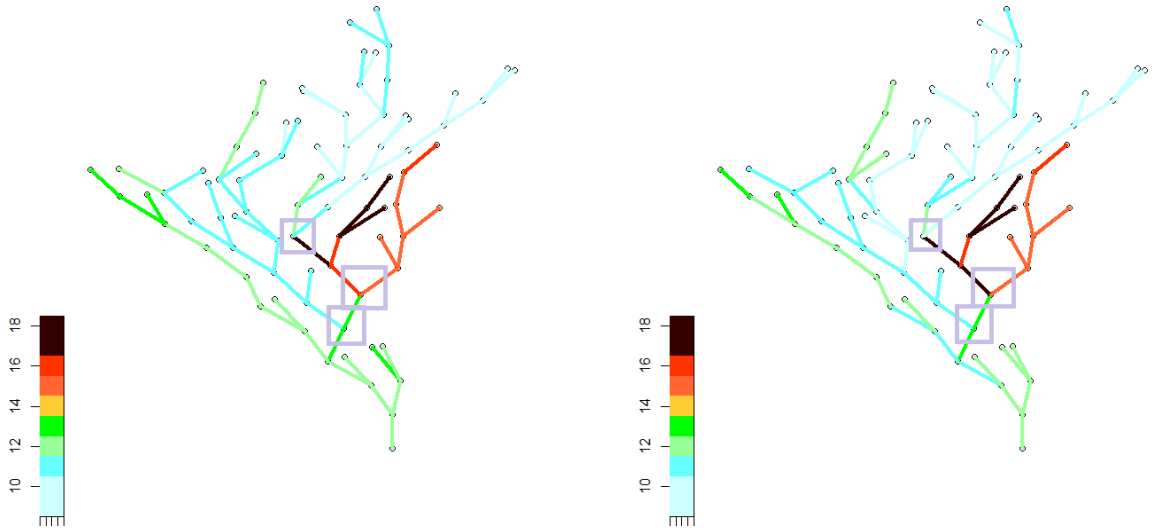

Figure 5: **Left:** The denoised river flow data by the nondecimated lifting algorithm 'LG-Aid-p-nlt' of Cao et al. (2024), using 30 trajectories. **Right:** The denoised river flow data by the proposed nondecimated lifting algorithm 'Bio-Haar-nlt-random' with 30 trajectories.

all (decimated) results, and BH is an extremely close competitor. However, when introducing nondecimation, BH becomes the optimal choice and leads to significant improvements for the performance of our E-LOCAAT algorithms. Compared with the line-graph based methods, our proposed BH demonstrates superior performance for the detection of discontinuities (see highlighted rectangles in Figure 5), as well as for recovering the edge function values at the boundary.

Both our proposed E-LOCAAT and competitor LG-LOCAAT methods with only *one* trajectory yield better results than the nondecimated method of Park and Oh (2022) using 50 trajectories.

4.2 Real data analysis: road traffic estimation

To show the performance of our algorithm for a more compelling real-life problem, we also analyse the edge data collected from a larger dense network. The Chicago-Sketch network structure, introduced by LeBlanc et al. (1975) and illustrated in Figure 1, consists of 933 nodes and 1475 edges, along with the corresponding traffic volume and generalised cost data.

We perform the nondecimated variants of the 'Bio-Haar' algorithm (BH) and its competitor

'LG-Aid' (LG), both with 100 trajectories. Note the Chicago-Sketch network structure is highly irregular in terms of edge lengths, for example, many short edges are adjacent to some relatively long edges, particularly along the network boundary. Thus, if an edge with a short length is picked to be predicted, and it has only one neighbouring edge with a relatively large length, then the update coefficients will be close to one and potentially cause a stability issue, as discussed in Appendix F. Hence in order to give a robust and well-behaved estimator, we take the median of the estimates from the P (randomly sampled) trajectories.

Figure 6 comparatively illustrates the denoising performance of the two methods for the traffic volume data. Compared to BH, the line-graph algorithm is more likely to smooth a larger area, see for example the edges around the network centre in the top row. The estimated signal-to-noise ratios of the denoised signal (equation (J.20)) are $\widehat{\text{SNR}}^{BH}_{vol,CS} \approx 3.2439$ and $\widehat{\text{SNR}}^{LG}_{vol,CS} \approx 2.5624$, indicating that the proposed BH captures more information from the observed data.

Figure 1 (right) shows that more areas of the cost dataset exhibit high variation, potentially indicating discontinuities in the underlying true function, and Figure 7 shows the denoised signal. For denser networks, LG-LOCAAT is less stable than E-LOCAAT (Appendix G.1) and may tend to oversmooth the underlying function. Indeed, we note that LG completely changes the pattern of the observations (top row), while BH has the ability to extract features in these regions. The SNR estimates are $\widehat{\text{SNR}}^{BH}_{cost,CS} \approx 1.9466$ and $\widehat{\text{SNR}}^{LG}_{cost,CS} \approx 0.6171$, which, corroborated with visualisations suggest that BH outperforms LG. The Q-Q plots (bottom row of Figure 7) show both sets of residuals have similar shapes and exhibit some departures from Gaussianity, but BH performs better in terms of the residual magnitude. Analysis of a smaller road network (Sioux Falls, Appendix J) shows similar significant noise reduction.

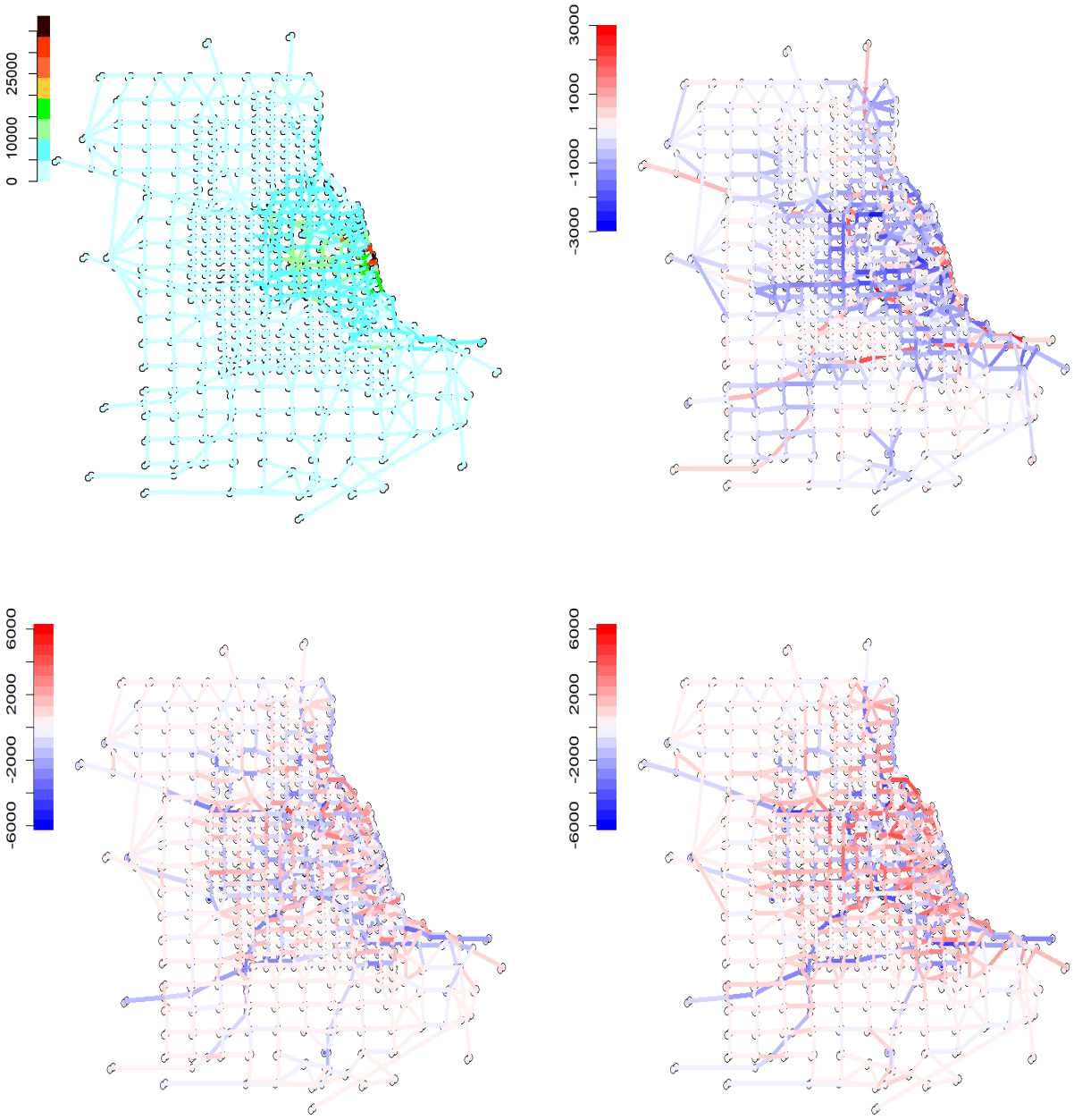

Figure 6: Denoising results for the Chicago-Sketch traffic volume data via the nondecimated variants of proposed 'Bio-Haar' algorithm (BH) and its competitor 'LG-Aid'. **Top Left:** Denoised volume by BH (100 trajectories). **Top Right:** Difference of the performance between BH (100 trajectories) and 'LG-Aid' (100 trajectories). **Bottom Left:** Residual plot (BH). **Bottom Right:** Residual plot ('LG-Aid').

Data Availability Statement

The network associated to the simulated flow dataset in Section 4 can be obtained using openly available code in the supplementary material of Park and Oh (2022). The traffic data for the

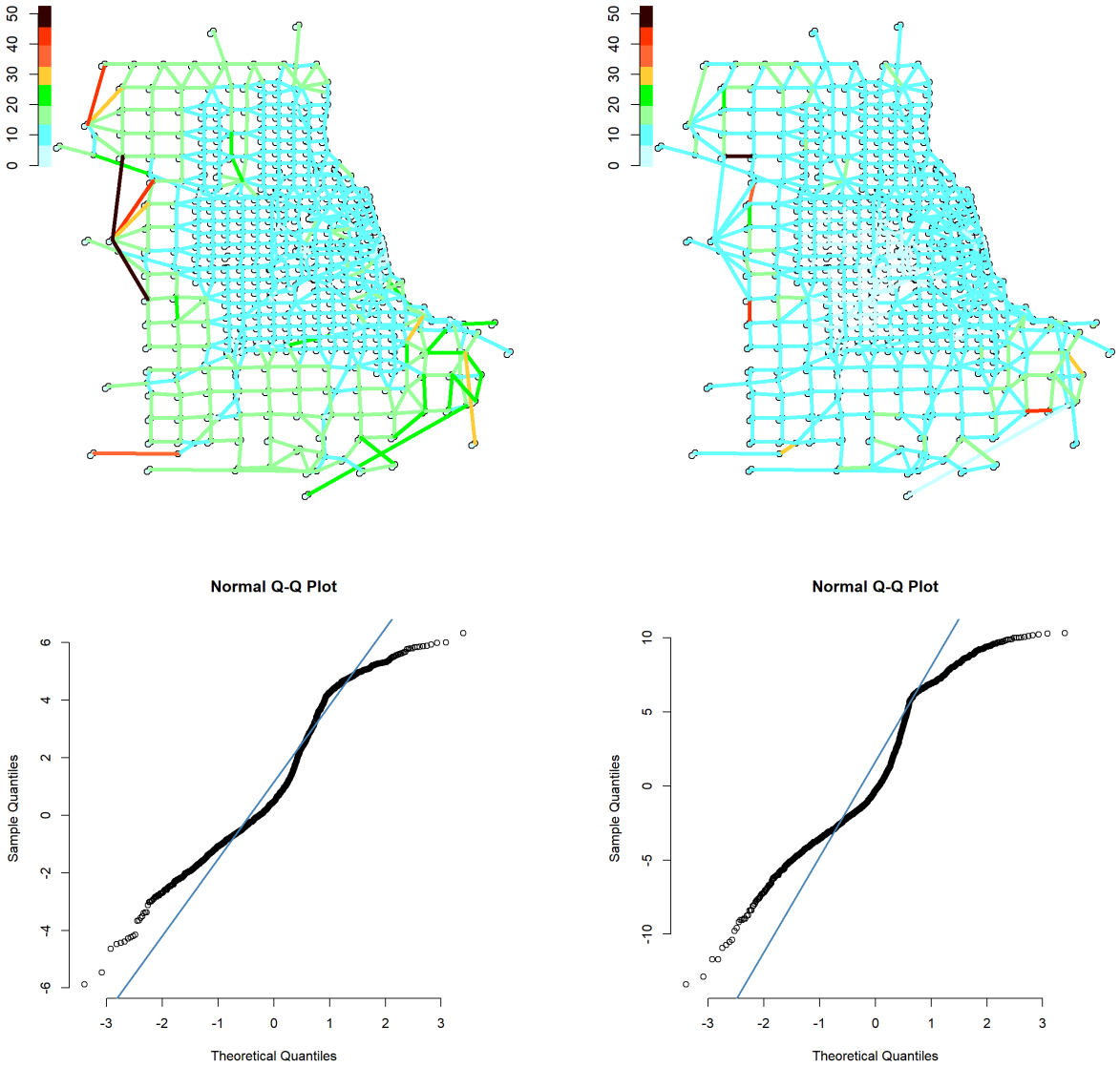

Figure 7: Denoising results for the Chicago-Sketch traffic cost data via the nondecimated variants of proposed 'Bio-Haar' algorithm (BH, left) and its competitor 'LG-Aid' (right), both with 100 trajectories. **Top row:** Denoised cost. **Bottom row:** Residual Q-Q plot.

Chicago-Sketch and Sioux Falls road networks are openly available in GitHub at

https://github.com/bstabler/TransportationNetworks.

Acknowledgments

MIK and MAN gratefully acknowledge support from EPSRC NeST Programme Grant EP/X002195/1.

SUPPLEMENTARY MATERIAL

Appendix: Contains further theoretical arguments and supporting simulation results. (pdf file)

References

Baker, M. and X. Faber (2006). Metrized graphs, Laplacian operators, and electrical networks. *Contemporary Mathematics 415*(2), 15–34.

Boginski, V., S. Butenko, and P. M. Pardalos (2005). Statistical analysis of financial networks. *Computational Statistics & Data Analysis 48*(2), 431–443.

Bollobás, B. (1998). *Modern Graph Theory*, Volume 184 of *Graduate Texts in Mathematics*. Springer Science & Business Media.

Bondy, J. A. and U. S. R. Murty (1976). *Graph Theory with Applications*. Elsevier Science Ltd.

Bullmore, E. and O. Sporns (2009). Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience 10*(3), 186–198.

Buzenkov, I. I., Y. V. Redkin, and A. A. Tyufanova (2024). Modeling methods for smoothing signals of wireless sensor networks. In *AIP Conference Proceedings*, Volume 3183, 070002. AIP Publishing.

Cao, D., M. I. Knight, and G. P. Nason (2024). A multiscale method for data collected from network edges via the line graph. *arXiv preprint arXiv:2410.13693*.

Doreian, P. and N. Conti (2012). Social context, spatial structure and social network structure. *Social Networks 34*(1), 32–46.

Dörfler, F., J. W. Simpson-Porco, and F. Bullo (2018). Electrical networks and algebraic graph theory: Models, properties, and applications. *Proceedings of the IEEE 106*(5), 977–1005.

Gallacher, K., C. Miller, E. Scott, R. Willows, L. Pope, and J. Douglass (2017). Flow-directed PCA for monitoring networks. *Environmetrics 28*(2), e2434.

Harary, F. (2018). *Graph Theory (on Demand Printing Of 02787)*. CRC Press, Boca Raton.

Hoef, J. M. V., E. Peterson, and D. Theobald (2006). Spatial statistical models that use flow and stream distance. *Environmental and Ecological Statistics 13*(4), 449–464.

Huttenlocher, D. P., G. A. Klanderman, and W. J. Rucklidge (1993). Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*(9), 850–863.

Jansen, M., G. P. Nason, and B. W. Silverman (2009). Multiscale methods for data on graphs and irregular multidimensional situations. *Journal of the Royal Statistical Society Series B 71*(1), 97–125.

Jawerth, B. and W. Sweldens (1994). An overview of wavelet based multiresolution analyses. *SIAM Review 36*(3), 377–412.

Johnstone, I. M. and B. W. Silverman (2004). Needles and straw in haystacks: Empirical Bayes estimates of possibly sparse sequences. *The Annals of Statistics 32*(4), 1594–1649.

Karimi, D. and S. E. Salcudean (2019). Reducing the Hausdorff distance in medical image segmentation with convolutional neural networks. *IEEE Transactions on Medical Imaging 39*(2), 499–513.

Kessels, F. (2019). *Traffic Flow Modelling*. EURO Advanced Tutorials on Operational Research, Series editors: M. Grazia Speranza and Jose Fernando Oliveira. Springer Cham.

Knight, M., M. Nunes, and G. Nason (2016). Modelling, detrending and decorrelation of network time series. *arXiv preprint arXiv:1603.03221*.

Knight, M. I. and G. P. Nason (2009). A 'nondecimated' lifting transform. *Statistics and Computing 19*(1), 1–16.

Knight, M. I., G. P. Nason, and M. A. Nunes (2017). A wavelet lifting approach to long-memory estimation. *Statistics and Computing 27*(6), 1453–1471.

Knight, M. I. and M. A. Nunes (2019). Long memory estimation for complex-valued time series. *Statistics and Computing 29*, 517–536.

Knight, M. I., M. A. Nunes, and G. P. Nason (2012). Spectral estimation for locally stationary time series with missing observations. *Statistics and Computing 22*, 877–895.

Kovac, A. and A. D. A. C. Smith (2011). Nonparametric regression on a graph. *Journal of Computational and Graphical Statistics 20*(2), 432–447.

Kuchment, P. (2003). Quantum graphs: I. some basic structures. *Waves in Random Media 14*(1), S107.

Kweon, S.-K. and K. G. Shin (2003). Statistical real-time communication over Ethernet. *IEEE Transactions on Parallel and Distributed Systems 14*(3), 322–335.

Lakhina, A., K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft (2004). Structural analysis of network traffic flows. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 61–72.

LeBlanc, L. J., E. K. Morlok, and W. P. Pierskalla (1975). An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research 9*(5), 309–318.

Lo Bello, L., G. Kaczynski, and O. Mirabella (2005). Improving the real-time behavior of ethernet networks using traffic smoothing. *IEEE Transactions on Industrial Informatics 1*(3), 151–161.

Mańka-Krasoń, A., A. Mwijage, and K. Kułakowski (2010). Clustering in random line graphs. *Computer Physics Communications 181*(1), 118–121.

Nacher, J., T. Yamada, S. Goto, M. Kanehisa, and T. Akutsu (2005). Two complementary representations of a scale-free network. *Physica A: Statistical Mechanics and its Applications 349*(1-2), 349–363.

Nason, G. P. and B. W. Silverman (1995). The stationary wavelet transform and some statistical applications. In *Wavelets and Statistics*, pp. 281–299. Springer.

Nunes, M. A., M. I. Knight, and G. P. Nason (2006). Adaptive lifting for nonparametric regression. *Statistics and Computing 16*(2), 143–159.

O'Donnell, D., A. Rushworth, A. W. Bowman, E. Marian Scott, and M. Hallard (2014). Flexible regression models over river networks. *Journal of the Royal Statistical Society Series C 63*(1), 47–63.

O'Searcoid, M. (2006). *Metric Spaces*. Springer Science & Business Media.

Park, S. and H.-S. Oh (2022). Lifting scheme for streamflow data in river networks. *Journal of the Royal Statistical Society Series C 71*(2), 467–490.

Schaub, M. T. and S. Segarra (2018). Flow smoothing and denoising: Graph signal processing in the edge-space. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 735–739. IEEE.

Schröder, P. and W. Sweldens (1995). Spherical wavelets: Efficiently representing functions on the sphere. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 161–172.

Schwock, F., J. Bloch, L. Atlas, S. Abadi, and A. Yazdan-Shahmorad (2023). Estimating and analyzing neural information flow using signal processing on graphs. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5.

Shuman, D. I., S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine 30*(3), 83–98.

Sweldens, W. (1996). The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis 3*(2), 186–200.

Sweldens, W. (1998). The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis 29*(2), 511–546.

Wang, X., S. Trajanovski, R. E. Kooij, and P. Van Mieghem (2016). Degree distribution and assortativity in line graphs of complex networks. *Physica A: Statistical Mechanics and its Applications 445*, 343–356.

Zhao, C., W. Shi, and Y. Deng (2005). A new Hausdorff distance for image matching. *Pattern Recognition Letters 26*(5), 581–586.