Article:
CAO, DINGJIA, KNIGHT, MARINA IULIANA and Nason, G.P. (2025) A multiscale method for data collected from network edges via the line graph. Statistics and computing. 200. ISSN: 0960-3174

https://doi.org/10.1007/s11222-025-10733-4

A multiscale method for data collected from network edges via the line graph

Dingjia Cao [1,*]· **Marina I. Knight**[1]· **Guy P. Nason**[2]

Abstract. Data collected over networks can be modelled as noisy observations of an unknown function over the nodes of a graph or network structure, fully described by its nodes and their connections, the edges. In this context, function estimation has been proposed in the literature and typically makes use of the network topology such as relative node arrangement, often using given or artificially constructed node Euclidean coordinates. However, networks that arise in fields such as hydrology (for example, river networks) present features that challenge these established modelling setups since the target function may naturally live on edges (e.g., river flow) and/or the node-oriented modelling uses noisy edge data as weights. This work tackles these challenges and develops a novel lifting scheme along with its associated (second) generation wavelets that permit data decomposition across the network edges. The transform, which we refer to under the acronym LG-LOCAAT, makes use of a line graph construction that first maps the data in the line graph domain. We thoroughly investigate the proposed algorithm's properties and illustrate its performance versus existing methodologies. We conclude with an application pertaining to hydrology that involves the denoising of a water quality index over the England river network, backed up by a simulation study for a river flow dataset.

[1] Department of Mathematics, University of York, York, UK

[2] Department of Mathematics, Imperial College London, UK

[*] Corresponding author: dingjia.cao@york.ac.uk

1 Introduction

Networks, as collections of interconnected objects mapped onto graph structures, allow us to understand the behaviour of complex systems and are relevant for many fields amongst which climate and hydrology are the focus of this work. Access to the network architecture has the potential to capture and model irregular data structures, for example, social and economic processes (Jackson and Watts, 2002); brain networks (Bullmore and Sporns, 2009); epidemiological networks (Danon et al., 2011); and traffic networks (Zhang et al., 2022). In recent years, the statistical modelling of networks has increasingly gained momentum, with applications such as subgraph density estimation (Chang et al., 2022); nonparametric regression on networks (Severn et al., 2021); spatio-temporal modelling on networks (Knight et al., 2020).

Although the current literature is rich when data is collected from the network's vertex space, in reality, data may naturally come from the edge set rather than the vertex set, for example, traffic

flow data (Lakhina et al., 2004); data from river networks (Cressie et al. 2006; Park and Oh 2022); and fish species distribution (Buisson et al., 2008).

Data collected from the edges of a metrized graph have been the focus of much statistical research, for example Cressie et al. (2006) and Hoef et al. (2006) introduced methods for network interpolation on edges using stream distance (similar to the shortest path distance in Baker and Faber (2006)), often associated with a high computational burden. O'Donnell et al. (2014) used nonparametric flexible regression models to analyze the data collected over river networks. Hoef et al. (2006) illustrate a stream distance limitation when used instead of Euclidean distance, in that it does not always produce a valid spatial covariance. Anderes et al. (2020) developed classes of covariance functions on linear graphs. Bolin et al. (2023a, 2024) introduced a new process class, the Gaussian Whittle-Matérn fields on metric graphs. Chaudhuri et al. (2025) apply these methods to traffic and environmental data, demonstrating their performance.

In our work we introduce a new approach capable to deal with the multiscale analysis of data collected from graph edges, affording competitive computation and denoising performance when evaluated against current literature methods. Let us next introduce a hydrological dataset that motivates our work.

1.1 Data collected from the network edges: a hydrological example

Hydrological data hold the key to ultimately understanding and/or forecasting related processes and their associated changes (McMillan et al., 2018). In this work, we focus on the dissolved oxygen (DO) measured by the amount of oxygen in the water, which is an indicator of water quality. Understanding DO data levels in their geographical context in turn allows us to assess the influence of weather and human behaviour (e.g., seasonal changes and pollution, respectively) on river ecology, beneficial from societal and economic perspectives.

Figure 1 provides a visualisation of DO levels over the river network in England. The river network can be separated into 10 river basins, and the 60 sampled stations are distributed in 9 out of 10 river basins. The size of the red circles is determined by the DO data values, where a larger circumference indicates a larger value. The dataset can be found in an open-resource website (https://environment.data.gov.uk/hydrology/explore).

Naturally, it would be desirable to consider the river system as a network structure, where the river conjunctions, sources and mouths can be considered as the vertices, and each river will be recognised as an edge. Since the data collecting stations are next to (or close to) corresponding rivers, it is more reasonable to consider the data as collected from network edges instead of vertices.

1.2 Aim and structure of the paper

As hydrological data measurements are typically noise contaminated (see McMillan et al. (2018) on imprecise measurements and data management errors), framing statistical modelling as nonparametric regression problems is highly valuable for the understanding of hydrological processes.

When compared with the analysis of data collected from the network vertex space, the analysis of data collected from network edges calls for new techniques, and in its turn reaches across many application fields. Wavelets, as a set of tools that allow us to capture the features of underlying functions in a multiscale fashion, are a desirable tool for performing the statistical analyses on hydrological processes. Due to the irregular nature of the network-structured data, the lifting scheme

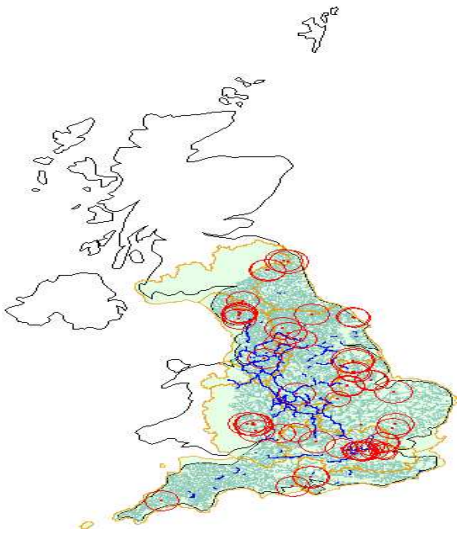

Fig. 1: England river network geometry (data collected on 10/05/2024). The green-coloured areas bounded by orange curves are different river basins. The light-blue grey curves are the river water bodies, and the blue ones are canal water bodies. The red dots are the stations that collected data, and the red circle circumference indicates the associated DO data value (larger circles indicate larger DO values).

introduced by Sweldens (1996, 1998) can be the key to such analyses. However, it is not trivial to apply the existing methods to edge-based data (see the heavy adaptation in Park and Oh (2022)), due to edge topology/geometry considerations, as opposed to the much simpler case when data have been collected over one-dimensional locations, or even over the vertex spaces (Jansen et al., 2009). Similarly, wavelets constructed on the vertex set of a network structure, referred to as the 'graph wavelets' among the signal processing community (Shuman et al., 2016; Stanković et al., 2020), cannot be directly employed on network edges.

Our contribution is a new multiscale methodology capable of denoising corrupted data collected from the network edges. The proposed method involves a transform based on the line graph technique, followed by a variant of the lifting one coefficient at a time (LOCAAT) scheme of Jansen et al. (2009) operating in the line graph space. Our proposed method allows for the construction of wavelet functions as well as for multiscale function representations, and has competitive denoising performance when compared with current literature methods that adjust existing vertex-based methodology for use on edges, such as Park and Oh (2022).

The paper is organised as follows. Section 2 gives a brief literature review of the key concepts that set the scene for introducing the proposed algorithm and its properties in Section 3 (see also Appendices A and B). Section 4 sets up a thorough simulation study that explores aspects such as algorithm sparsity, stability and its denoising properties (Appendix C gives further details). Section 5 investigates the DO dataset along with a simulated flow dataset previously introduced in the literature, and Section 6 concludes the work.

2 Background

In this section we introduce background knowledge for our methodology, ranging from graph theory to a variant of the wavelet lifting scheme.

2.1 Graphs and metrized graphs

Mathematically, a graph G can be represented as an ordered pair $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, ..., v_n\}$ is the set of vertices and $\mathcal{E} = \{e_1, ..., e_m\}$ is the set of edges, see e.g., Bondy et al. (1976). The cardinality of the vertex and the edge sets is $|\mathcal{V}| = n$ and $|\mathcal{E}| = m$, respectively. If the k-th edge indicates the connection between i-th and j-th vertices, then we write $e_k = \{v_i, v_j\}$, which is an unordered pair of these two vertices ($\{v_i, v_j\} = \{v_j, v_i\}$). In this case, we say v_j is a neighbour of v_i, and vice versa. We denote the

set of neighbouring vertices of the vertex v_i by $\mathcal{N}_i^{\mathcal{V}} = \{v_j \,|\, v_j \in \mathcal{V};\, \{v_i, v_j\} \in \mathcal{E}\}$.

A **weighted graph** G^ω is modelled by an ordered pair (G, ω), where $\omega : \mathcal{E} \longrightarrow \mathbb{R}$ is a function that associates a weight to each edge (Bondy et al., 1976). To fit our framework, we let $\omega : \mathcal{E} \longrightarrow \mathbb{R}_+^\star$ for any undirected graph and for the edge $e_k = \{v_i, v_j\} \in \mathcal{E}$, the notation ω_k or ω_{ij} will be used interchangeably to represent the weight defined on this edge. In the literature on graph theory, there are many different ways to consider the weights associated with edges; for example, weights can be the cost of the travelling salesman problem (Bondy et al., 1976), resistance in electronic networks (Bollobás, 2013), or traffic flow in a road network. The size of the weights has different (sometimes opposite) effects in various applications, e.g., a large weight on an edge indicates a weak connection between the vertices when the weights represent cost, while in the case of traffic flow, it indicates a strong link.

For the purpose of our work, we let the connection between two vertices v_i and v_j increase monotonically with the weight of the edge $e_k = \{v_i, v_j\}$. Therefore, if v_i and v_j are close to each other, then the weight ω_k tends to be large. Weighted graphs are widely used in many classical graph theory problems. However, recently, more focus has been put on metric graphs, due to their wide ranging applications, for example, for Hilbert spaces defined on graphs (Kuchment, 2003) and for the analysis of electronic networks (Baker and Faber, 2006). In a nutshell, the metrized graph gives geometric properties for a weighted graph and allows us to define and analyse functions in the graph domain. A metrized graph Γ of a weighted graph $G^\omega = (G, \omega)$ arises from a pair (G, ℓ), where $\ell : \mathcal{E} \longrightarrow \mathbb{R}_+$ is a function that assigns lengths to all edges, in the following way. Here we adopt the construction from Baker and Faber (2006) and define the length as inverse weight, such that $\ell(e) = 1/\omega(e)$, for all $e \in \mathcal{E}$. Then to each edge e, we associate a line segment (or equivalently, a one-dimensional interval) of length $\ell(e)$ and identify the ends of distinct line segments if they correspond to the same vertex $v \in \mathcal{V}$. For the length of a certain edge, say the k-th edge e_k, we write ℓ_k for convenience. The space $\Gamma(G^\omega)$ is the space that contains the points in any of these line segment intervals, and G^ω is referred to as a model for $\Gamma(G^\omega)$ (simply Γ if there is no ambiguity). The distance between two points in Γ is defined as the length of the shortest path between them along the line segments traversed. This distance is referred to as the path metric, and Γ with this metric form a metrized graph.

Baker and Faber (2006) also introduced a way to define a vertex set and the associated edge set on Γ, summarised below.

- **Defining a vertex set on Γ:**
  Let $\mathrm{Vex}(\Gamma)$ be any finite, non-empty subset of Γ, such that $\mathrm{Vex}(\Gamma)$ contains all points with $n_{\mathbf{p}} \neq 2$, where $n_{\mathbf{p}}$ denotes the number of the directions by which a path can leave the point $\mathbf{p}$, which is referred to as the valence in Baker and Faber (2006). Hence, $\Gamma \backslash \mathrm{Vex}(\Gamma)$ is a finite, disjoint union of subspaces $\{U_k\}_k$ isometric to open intervals, where U_k can be considered as a neighbourhood of a point $\mathbf{p}_k \in \Gamma$.
- **Defining an edge set on Γ:**
  Once a vertex set has been defined, the associated (metrized) edge set can be constructed based on it. We define the topological closure of U_k, isometric to a line segment, to be $e_k^{\mathrm{met}} = \overline{U}_k$ as the k-th edge of the metrized graph. Alternatively, denote the metrized k-th edge e_k^{met} of $e_k = \{v_i, v_j\}$ as $[v_i, v_j]$ for convenience. Additionally, any distinct metrized edges e_k^{met} and $e_{k'}^{\mathrm{met}}$ $(k \neq k')$ intersect in at most one point.

For a metrized graph Γ corresponding to the original graph G^ω, the space $L^2(\Gamma)$ (Kuchment,

2003) consists of functions that are measurable and integrable on each metrized edge, such that

$$\|f\|^2_{L^2(\Gamma)} = \sum_{e\in\mathcal{E}} \|f\|^2_{L^2(e^{\text{met}})} < \infty,$$

where the metrized edge e^{met} corresponding to the original edge $e \in \mathcal{E}$ is isometric to a line segment $[0, \ell(e^{\text{met}})]$ and $\ell(e^{\text{met}})$ is the length of this edge.

The space $L^2(\Gamma)$ can be considered as the orthogonal direct sum of $L^2(e^{\text{met}})$, see Kuchment (2003). Consequently, since e^{met} is isometric to a closed interval on $\mathbb{R}$, the inner product and orthogonality can be defined as in $L^2(\mathbb{R})$.

2.2 A wavelet lifting transform for graphs

Our algorithm will make use of the LOCAAT (**L**ifting **O**ne **C**oefficient **A**t **A** **T**ime) transform of Jansen et al. (2009), which is a variant of the lifting scheme (Sweldens, 1998) that performs a split stage that isolates one point at each step, as opposed to carrying out an odds and evens split. Assuming the graph is underpinned by a metric such as path length or Euclidean distance, LOCAAT provides a wavelet-like decomposition that only relies on the network topology and on the inter-vertex distance information.

For a function $f^{\mathcal{V}} : \mathcal{V} \to \mathbb{R}$, denote by $f_i^{\mathcal{V}}$ the observation at the i-th vertex $v_i \in \mathcal{V}$, for all $i \in \{1, ..., n\}$. The initial scaling coefficients are obtained by taking $c^{\mathcal{V}}_{i,n} := f_i^{\mathcal{V}}$, for all $i \in \{1, ..., n\}$. Then carry out the stage-n LOCAAT transform, consisting of four steps as follows.

- **Split:** One vertex, namely v_{i_n}, will be chosen to be predicted and then to be removed.
- **Predict:** The observation value $f^{\mathcal{V}}_{i_n}$ associated with v_{i_n} will be predicted using its neighbouring nodes ($\mathcal{N}^{\mathcal{V}}_{i_n,n}$), and the residue denoted by

$$d^{\mathcal{V}}_{i_n} = c^{\mathcal{V}}_{i_n,n} - \sum_{j:\, v_j\in\mathcal{N}^{\mathcal{V}}_{i_n,n}} a^{\mathcal{V}}_{j,n} c^{\mathcal{V}}_{j,n}, \qquad (1)$$

where $\{a^{\mathcal{V}}_{j,n}\}_{j:\, v_j\in\mathcal{N}^{\mathcal{V}}_{i_n,n}}$ is the prediction filter at stage-n. The difference $d^{\mathcal{V}}_{i_n}$ between the observation value and the prediction value is interpreted as the detail (wavelet) coefficient.

- **Update:** The scaling coefficients associated with the neighbouring vertices of v_{i_n} will be updated from stage-n to stage-$(n-1)$ as follows

$$c^{\mathcal{V}}_{j,n-1} = c^{\mathcal{V}}_{j,n} + b^{\mathcal{V}}_{j,n} d^{\mathcal{V}}_{i_n}, \ \ \forall j \in \mathcal{N}^{\mathcal{V}}_{i_n,n}, \qquad (2)$$

with $\{b^{\mathcal{V}}_{j,n}\}_{j\in\mathcal{N}^{\mathcal{V}}_{i_n,n}}$ the stage-n update filter, and the remaining scaling coefficients are kept the same. Thus, $\forall k \notin \mathcal{N}^{\mathcal{V}}_{i_n,n}$, we let $c^{\mathcal{V}}_{k,n-1} := c^{\mathcal{V}}_{k,n}$. In addition, the neighbouring integral values will also be updated to account for the loss of vertex v_{i_n}, such that

$$I^{\mathcal{V}}_{j,n-1} = I^{\mathcal{V}}_{j,n} + a^{\mathcal{V}}_{j,n} I^{\mathcal{V}}_{i_n,n}, \ \ \forall j \in \mathcal{N}^{\mathcal{V}}_{i_n,n}. \qquad (3)$$

For $\forall k \notin \mathcal{N}^{\mathcal{V}}_{i_n,n}$, we let $I^{\mathcal{V}}_{k,n-1} := I^{\mathcal{V}}_{k,n}$. The lifting scale, a generalisation of the classical wavelet scale notion, is taken to be the integral associated to each vertex at its removal stage, here $I^{\mathcal{V}}_{i_n,n}$. Note that the lifting scale has a continuous, rather than discrete character, useful in a variety of statistical problems outside the realm of nonparametric regression (Knight et al., 2017).

- **Relink:** Once the vertex v_{i_n} has been removed, Jansen et al. (2009) suggest performing a minimum spanning tree among the neighbouring node set $\{v_j \in \mathcal{N}^{\mathcal{V}}_{i_n,n}\}$ for relinkage.
- **Iterate :** Repeat the steps above until a stopping time τ set in advance. Typically, iterate the above steps $(n-2)$ times, resulting in $(n-2)$ detail coefficients and $\tau = 2$ scaling coefficients. At each stage-r, the split-predict-update-relink

procedure is of the same form as the one presented above (stage-n), and we only change notation n to r, e.g., i_r indicates the vertex chosen for removal at stage-r.

Now taking a function representation angle, typical in wavelet-based applications, at the initial stage-n, the observed function is approximated by a linear combination of scaling functions, namely

$$f^{\mathcal{V}}(v) = \sum_{i=1}^{n} c_{i,n}^{\mathcal{V}} \varphi_{i,n}(v), \tag{4}$$

where $\{\varphi_{i,n}\}_i$ are interpolating (primal) scaling functions that fulfill the property $\varphi_{i,n}(v_j) = \delta_{i,j}$, with $\delta_{i,j}$ denoting the Kronecker delta, and the initial scaling coefficients are $c_{i,n}^{\mathcal{V}} := \langle \tilde{\varphi}_{i,n}, f^{\mathcal{V}} \rangle = f_i^{\mathcal{V}}$. The interpolating property of the primal scaling functions and their biorthogonality with the dual scaling functions $\{\tilde{\varphi}_{i,n}\}_i$, such that $\langle \tilde{\varphi}_{i,n}, \varphi_{j,n} \rangle = \delta_{i,j}$, are typical conditions imposed for second generation wavelet decompositions (Sweldens, 1998) and ensured for example by starting with a set of characteristic functions associated with a graph partition and the Dirac delta functions as primal and dual bases, respectively (Jansen et al., 2009). The function representation setup informs the choices in the split-predict-update steps. In the split step, we mark for removal vertices in the denser sampled area of the graph. These are decided upon by taking into account the integral of the primal scaling functions, $\{I_{i,n}^{\mathcal{V}}\}_{i:\, v_i \in \mathcal{V}}$, obtained e.g., by using a vertex-associated graph partitioning then removing the vertex with the smallest integral, v_{i_n}. In the classical spirit of wavelet properties, the predict step is designed such that ideally the detail (wavelet) coefficient associated with a vertex is zero should the function be constant in its neighbourhood; this yields the prediction weights property $\sum_{j:\, v_j \in \mathcal{N}_{i_n,n}^{\mathcal{V}}} a_{j,n}^{\mathcal{V}} = 1$. The predict step is also responsible for building a new set of scaling functions, associated to the next stage-$(n-1)$, $\{\varphi_{i,n-1}\}_{i \neq i_n}$, as well as a new wavelet function ψ_{i_n} (Jansen et al., 2009, Section 2.3). The usual oscillatory wavelet property ensuring zero integral of ψ_{i_n} in turn induces the update filter constraint $\sum_{j:\, v_j \in \mathcal{N}_{i_n,n}^{\mathcal{V}}} b_{j,n}^{\mathcal{V}} I_{j,n-1} = I_{i_n,n}$. At the start of stage-$(n-1)$, the new primal basis is $\{\varphi_{i,n-1}, \psi_{i_n}\}_{i \neq i_n}$, and a corresponding biorthogonal dual basis is also constructed. It is on these new bases that the function is next represented, and reiteration of the algorithm yields at stage-r

$$f^{\mathcal{V}}(v) = \sum_{i \in \mathcal{S}_r} c_{i,r}^{\mathcal{V}} \varphi_{i,r}(v) + \sum_{i \in \mathcal{D}_r} d_i^{\mathcal{V}} \psi_i(v), \tag{5}$$

with $\mathcal{S}_r$ and $\mathcal{D}_r$ denoting the sets of scaling and wavelet coefficients, respectively. For a dyadic level split akin to that of classical wavelets, Jansen et al. (2009) propose to group the wavelet coefficients associated to similar (continuous) scales into 'artificial' levels by a quantile split of the monotonically arranged collection $\{I_{i,r}^{\mathcal{V}}\}_{i \in \mathcal{D}_r}$, with the finest level corresponding to the lowest values.

The LOCAAT transform is invertible through iteratively undoing each of its steps, or equivalently by noticing that being a linear transform, it can be expressed using matrix multiplication (Nunes et al., 2006).

2.3 Nonparametric regression over graphs

In general terms, nonparametric regression problems amount to typically modellling the observations $\{(x_i, f_i)\}_{i=1}^n$ as being additively contaminated by noise $\{\epsilon_i\}_{i=1}^n$

$$f_i = g(x_i) + \epsilon_i, \tag{6}$$

where $x_i \in \Omega$ is the data location from a certain bounded domain Ω, and $g : \Omega \longrightarrow \mathbb{R}$ is a true, unknown function that we aim to estimate. The noise is assumed to be a set of independent, identically distributed (iid) normal random

variables, $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$, where σ^2 is finite and usually unknown. The bounded domain can be a one-dimensional interval $\Omega = [0, 1]$ (Nunes et al., 2006), high-dimensional Euclidean space, or graph vertex domain $\Omega = \mathcal{V}$ for a graph $G = (\mathcal{V}, \mathcal{E})$ (Jansen et al., 2009; Mahadevan, 2010).

The classical LOCAAT transform introduced in Section 2.2 can be used for denoising the set of nodal observations $\underline{f}^{\mathcal{V}} = \{f_i^{\mathcal{V}}\}_{i=1}^n$, by first projecting them into the lifting domain and obtaining a set of wavelet coefficients $\underline{d}^{\mathcal{V}}$. To this end, an empirical Bayes thresholding approach adapted to the lifting domain can be taken (Johnstone and Silverman, 2004; Nunes et al., 2006), followed by inverting LOCAAT. This leads to obtaining an estimator $\underline{\hat{g}}^{\mathcal{V}}$ for the true (unknown) function $g^{\mathcal{V}}$, where the superscript indicates the fact that the observations, and consequently the estimates, are obtained over the network vertices.

For the problem tackled through this work, the observations are collected over the graph edges $\Omega = \mathcal{E}$, and a direct application of LOCAAT for projection in the wavelet domain is not possible.

3 Proposed LG-LOCAAT algorithm

We introduce an approach to carry out multiscale analysis on data collected from the edges of a network modelled as a graph $G = (\mathcal{V}, \mathcal{E})$. Denote a set of observations on a function $g^{\mathcal{E}}$ collected from the graph edges by $\{g_k^{\mathcal{E}}\}_{k=1}^m$. The method we propose involves initially applying a line graph transform in order to shift emphasis from (original) edges to (new) vertices, followed by a version of the LOCAAT on the vertices in the line graph domain. Therefore, we refer to our proposed technique as **L**ine **G**raph **LOCAAT** (LG-LOCAAT).

Specifically, the line graph of a graph G, denoted $\mathbf{LG}(G)$, is the graph whose vertex set is bijective to the edge set $\mathcal{E}$ of G. For convenience, we write $\mathbf{LG}(G) = G^* = (\mathcal{V}^*, \mathcal{E}^*)$, and we have $\mathcal{V}^* \longleftrightarrow \mathcal{E}$, where '$\longleftrightarrow$' indicates a bijection between two sets. Moreover, we simply let a new vertex v_k^* correspond to the k-th original edge $e_k \in \mathcal{E}$, for any $k \in \{1, ..., |\mathcal{E}|\}$. Although the correspondence of subscripts can be any permutation, it will not affect the result of our work since LOCAAT does not rely on this ordering. The new edge set $\mathcal{E}^*$ can be defined as

$$\mathcal{E}^* = \{\{v_k^*, v_l^*\} \,|\, e_k, e_l \in \mathcal{E} \text{ and } |e_k \cap e_l| = 1\}\,.$$

Here the cardinality of the intersection of two (original) edges being equal to one indicates that e_k and e_l share exactly one common vertex in the graph G. Similarly, the neighbourhood $\mathcal{N}_k^{\mathcal{V}^*}$ of a vertex v_k^* in the line graph is defined as the set of new vertices which are connected with v_k^* by an edge $e^* \in \mathcal{E}^*$, mathematically represented as

$$\mathcal{N}_k^{\mathcal{V}^*} := \{v_s^* \in \mathcal{V}^* \,|\, \{v_k^*, v_s^*\} \in \mathcal{E}^*\} \tag{7}$$

$$= \{v_s^* \in \mathcal{V}^* \,|\, s \neq k;\, e_k \cap e_s \neq \emptyset;\, e_k, e_s \in \mathcal{E}\} \tag{8}$$

Note that line graphs reveal only combinatorial information between edges. To perform LOCAAT on a line graph, we will use the path length or Euclidean distances between the new vertices.

Before introducing our proposed algorithm, let us first take a function representation perspective. Following the notation of Diestel (2005), suppose that we have a real-valued function $g^{\mathcal{E}} \in \mathfrak{E}$, where $\mathfrak{E}$ is the vector space that contains all functions that map $\mathcal{E}$ to $\mathbb{R}$, where $\mathcal{E}$ is the edge set of the original graph G. Similarly, a vector space $\mathfrak{V}^*$ containing all functions defined on the new vertex set of the line graph $\mathbf{LG}(G) = G^*$ can be also defined. The bijection between $\mathcal{E}$ and $\mathcal{V}^*$ indicates that the two vector spaces $\mathfrak{E}$ and $\mathfrak{V}^*$ are isomorphic, denoted as $\mathfrak{E} \approx \mathfrak{V}^*$ (Roman et al., 2005). Thus, there exists a function $g^{\mathcal{V}^*} \in \mathfrak{V}^*$, where

$g^{\mathcal{V}^*}(v_k^*) = g^{\mathcal{E}}(e_k)$ if v_k^* is the image of e_k according to the line graph transform. We denote $g^{\mathcal{V}^*} \equiv g^{\mathcal{E}}$ if they satisfy this condition for all k. This isomorphism has many advantages for our construction. First of all, since additivity and scalar multiplication are maintained in isomorphic vector spaces (Roman et al., 2005), performing a linear combination of a set of vectors $\{\mathbf{u}_s \in \mathfrak{V}^*\}_s$ is equivalent to performing the same linear combination of a set of vectors $\{\mathbf{w}_s \in \mathfrak{E}\}_s$, where $\mathbf{u}_s \equiv \mathbf{w}_s$. Secondly, if $\{\mathbf{u}_s\}_{s=1}^p$ is a basis defined on $\mathfrak{V}^*$, then $\{\mathbf{w}_s\}_{s=1}^p$ is a basis defined on $\mathfrak{E}$, and $\mathbf{u}_s \equiv \mathbf{w}_s$. As a result, it is feasible to design a multiresolution analysis (MRA) for $g^{\mathcal{E}}$ based on $\mathcal{V}^*$.

Incorporating the results from Sweldens (1996, 1998) and the LOCAAT framework, we summarise the conditions for suitably designing primal and dual scaling functions for the vertex set $\mathcal{V}^*$ of the line graph G^*, as follows. The set of dual scaling functions $\{\tilde{\varphi}_{k,m}^{\mathcal{V}^*}\}_{k=1}^m$ should satisfy $\langle \tilde{\varphi}_{k,m}^{\mathcal{V}^*}, g^{\mathcal{V}^*} \rangle = g_k^{\mathcal{V}^*}$, where $\{g_k^{\mathcal{V}^*}\}_{k=1}^m$ is the set of original edge observations, now mapped onto the line graph vertices. This condition allows us to start with the observation values as the initial scaling coefficients, and then to perform the lifting scheme. The primal scaling functions $\{\varphi_{k,m}^{\mathcal{V}^*}\}_{k=1}^m$ are designed such that for each $k \in \{1, ..., m\}$, we have $\varphi_{k,m}^{\mathcal{V}^*}(v_k^*) = 1$ and $\varphi_{k,m}^{\mathcal{V}^*}(v_s^*) = 0$ for all $s \neq k$. As for typical lifting constructions, the set of primal and dual initial scaling functions has to satisfy the biorthogonality property, such that $\langle \tilde{\varphi}_{k,m}^{\mathcal{V}^*}, \varphi_{k',m}^{\mathcal{V}^*} \rangle = \delta_{kk'}$.

For a graph G, we denote the associated metric space of its line graph by $(\mathcal{V}^*, \mathrm{dist}_{\mathcal{V}^*})$ and its metrized version by $(\Gamma^*, \mathrm{dist})$. The distance measure (dist) between two metrized vertices is defined as the length of the corresponding metrized edge $e_k^{*\mathrm{met}}$, for any edge $e_k^* = \{v_i^*, v_j^*\}$. We denote a point in the metrized graph space as $\mathbf{p}^* \in \Gamma^*$, and denote by $\mathbf{p}_{v_k^*}^*$ the point that corresponds to v_k^*. Then we can define a set of partitionings, $\{\mathbf{P}_s^*\}_{s=1}^m$, of the metrized line graph such that

$$\mathbf{P}_s^* = \Big\{ \mathbf{p}^* \in \Gamma^* \,|\, \mathrm{dist}(\mathbf{p}^*, \mathbf{p}_{v_s^*}^*) < \mathrm{dist}(\mathbf{p}^*, \mathbf{p}_{v_{s'}^*}^*), \\ \forall s' \in \{1, ..., m\} \backslash \{s\} \Big\}, \tag{9}$$

e.g., the middle points of the metrized edges $e^{*\mathrm{met}}$ generate a partitioning. In the same vein as Jansen et al. (2009), the $s = 1, ..., m$ primal and dual scaling functions can be defined as

$$\varphi_{s,m}^{\Gamma^*}(\mathbf{p}^*) = \chi_{\mathbf{P}_s^*}(\mathbf{p}^*);$$
$$\tilde{\varphi}_{s,m}^{\Gamma^*}(\mathbf{p}^*) = \delta(\mathbf{p}^* - \mathbf{p}_{v_s^*}^*),$$

where $\chi_{\mathbf{P}_s^*}$ is the characteristic function defined on the s-th block of the partitioning, and $\delta(\mathbf{p}^* - \mathbf{p}_{v_s^*}^*)$ is the Dirac delta with the energy centred on the point $\mathbf{p}_{v_s^*}^*$. (The Γ^* superscript indicates the scaling functions are defined on the metrized line graph space.) The advantage of the characteristic functions is that they can be used to reveal the geometric information of the partitionings. We also consider an alternative setup, namely

$$\varphi_{s,m}^{\Gamma^*}(\mathbf{p}^*) = \delta_{\mathbf{p}_{v_s^*}^*, \mathbf{p}^*}; \tag{10}$$
$$\tilde{\varphi}_{s,m}^{\Gamma^*}(\mathbf{p}^*) = \delta(\mathbf{p}^* - \mathbf{p}_{v_s^*}^*), \tag{11}$$

where $\delta_{\mathbf{p}_{v_s^*}^*, \mathbf{p}^*}$ is the Kronecker delta. The construction using equations (10) and (11) can be considered akin to the lazy wavelets introduced by Sweldens (1996), but on the new vertex set.

Then the initial expansion form for the function approximation can be written as

$$g^{\mathcal{E}}(e) \equiv g^{\mathcal{V}^*}(v^*) = g^{\Gamma^*}(\mathbf{p}^*) = \sum_{s=1}^{m} c_{s,m}^{\Gamma^*} \varphi_{s,m}^{\Gamma^*}(\mathbf{p}^*)$$

where $\mathbf{p}^* \in \Gamma^*$ and g^{Γ^*} is the metrized analogue of the function $g^{\mathcal{V}^*}$, and the (initial) scaling coefficients are $c_{s,m}^{\Gamma^*} := \langle g^{\Gamma^*}, \tilde{\varphi}_{s,m}^{\Gamma^*} \rangle = g_s^{\Gamma^*} = g_s^{\mathcal{E}}$.

From this point onwards, we use $g^{\mathcal{V}^*}$ interchangeably with g^{Γ^*}. The aim is to find the function approximation as follows

$$g^{\Gamma^*}(\mathbf{p}^*) = \sum_{s\in\mathcal{S}_2} c_{s,2}^{\Gamma^*}\varphi_{s,2}^{\Gamma^*}(\mathbf{p}^*) + \sum_{l\in\mathcal{D}_2} d_l^{\Gamma^*}\psi_l^{\Gamma^*}(\mathbf{p}^*), \tag{12}$$

where $\mathcal{D}_2 = \{k_m, ..., k_3\}$, and $\mathcal{S}_2 = \{1, ..., m\}\backslash\mathcal{D}_2$. The set $\{d_l^{\Gamma^*}\}_{l\in\mathcal{D}_2}$ holds the detail (wavelet) coefficients, which can be obtained at each stage-r along with the scaling coefficients $\{c_{s,r}^{\Gamma^*}\}_{s\in\mathcal{S}_r}$ by our proposed algorithm next detailed.

3.1 LG-LOCAAT algorithm

We now present our proposed algorithm in terms of the iterative split-predict-update-relink procedure on the set of graph edge observations $\{g_k^{\mathcal{E}}\}_{k=1}^m$. The choices we make are informed by previous graph LOCAAT findings (Jansen et al., 2009; Mahadevan, 2010) and adapted to the line graph domain. As discussed above, we map the edge observations $\{g_k^{\Gamma^*}\}_{k=1}^m$ into the metrized line graph domain. We start from stage-m, corresponding to the original edge-based observations, and the initial scaling coefficients are defined as $c_{k,m}^{\Gamma^*} := \langle g^{\Gamma^*}, \tilde{\varphi}_{k,m}^{\Gamma^*}\rangle = g_k^{\Gamma^*} = g_k^{\mathcal{E}}$. The initial neighbourhood structure can be represented in the line graph domain as described in equations (7) or (8). In what follows, we write $\mathcal{N}_{k,m}^{\mathcal{V}^*}$ instead of $\mathcal{N}_k^{\mathcal{V}^*}$ since the neighbourhood structure will be changed as the algorithm progresses through stage-m, stage-$(m-1)$, and so on.

- **Split:** In line with Jansen et al. (2009), choose the new vertex to be removed, denote it by $v_{k_m}^*$, according to minimum integral value for the primal scaling function, here associated with its metrized point $\mathbf{p}_{v_{k_m}^*}^*$. When the initial primal scaling functions are defined as characteristic functions on the partitionings of the metrized line graph, the initial integral values are

$$\begin{aligned} I_{k,m}^{\Gamma^*,\text{sum}} &= \int_{\Gamma^*} \varphi_{k,m}^{\Gamma^*}(\mathbf{p}^*)d\mathbf{p}^* \\ &= \int_{\Gamma^*} \chi_{\mathbf{P}_k^*}(\mathbf{p}^*)d\mathbf{p}^* \\ &= \mu(\mathbf{P}_k^*) \\ &\propto \sum_{s:v_s^*\in\mathcal{N}_{k,m}^{\mathcal{V}^*}} \text{dist}_{\mathcal{V}^*}(v_k^*, v_s^*), \end{aligned} \tag{13}$$

where we use the Lebesgue measure μ for a union of intervals (or line segments) to be the summation of their lengths, and '$\propto$' means 'proportional to'. The resulting integral is proportional to the sum of distances of the chosen new vertex to its neighbouring vertices. Computationally, using a proportional sum of distances will result in the same detail coefficients as when using the sum of distances, as shown by the following proposition, whose proof appears in Appendix B.

Proposition 1. *Suppose we have an integral sequence* $\underline{I}^* = \{I_{k,m}^*\}_{k=1}^m$, *and a constant* $C > 0$. *Then, performing the LOCAAT algorithm with* $C\cdot\underline{I}^*$ *as integrals will yield the same detail coefficients and the same prediction/update filters, as performing LOCAAT with* $\underline{I}^*$.

An alternative integral initialisation for the scaling functions was suggested by Jansen et al. (2009) to be the average distance, obtained as

$$I_{k,m}^{\Gamma^*,\text{ave}} = \frac{1}{2|\mathcal{N}_{k,m}^{\mathcal{V}^*}|} \sum_{s:v_s^*\in\mathcal{N}_{k,m}^{\mathcal{V}^*}} \text{dist}_{\mathcal{V}^*}(v_k^*, v_s^*). \tag{14}$$

We will also test the performance of the algorithm using the average distances as initial integrals since previous literature indicates its good performance (Mahadevan, 2010).
When the initial primal scaling functions are set as Kronecker delta, this leads to a variant of the lazy wavelets introduced in Sweldens (1998).

Denoting $\underline{\delta}_s = (0, \cdots, 0, 1, 0, \cdots, 0)$ as a canonical basis representation for v_s^*, such that only its s-th element is non-zero, we have

$$I_{k,m}^{\Gamma^*,\text{Delta}} = \langle \underline{\delta}_s, \mathbb{1}_m \rangle = 1, \tag{15}$$

where $\mathbb{1}_m$ is a vector of ones of length m.

Thus we will consider for the split step the following three integral value choices: sum of distances (equation (13), 'sum'), average distance (equation (14), 'ave'), and starting with a vector of ones (equation (15), 'Delta'). From now on, we will skip the superscript indicating the integral determination (sum/ave/Delta) unless necessary.

Once the initial integral values have been decided, we choose the new vertex to be predicted and then removed corresponding to the minimum integral value. If there exist multiple new vertices with the minimum integral value, then we randomly pick one of these vertices. As the LOCAAT framework follows the principle of recursive construction, we only need to fix the initial integrals and the recursive integral computation will be carried out through the iterative process, see Sweldens (1998). Therefore, in what follows, we refer to a general stage-r and its associated removal index, k_r.

- **Predict:** The detail coefficient obtained at stage-r is

$$d_{k_r}^{\Gamma^*} = c_{k_r,r}^{\Gamma^*} - \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} a_{s,r}^{\Gamma^*} c_{s,r}^{\Gamma^*}, \tag{16}$$

where $\{a_{s,r}^{\Gamma^*}\}_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}}$ are the prediction weights. The MRA framework associated with the prediction step can be recursively expressed as (Jansen et al., 2009),

$$\tilde{\psi}_{k_r}^{\Gamma^*} = \tilde{\varphi}_{k_r}^{\Gamma^*} - \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} a_{s,r}^{\Gamma^*} \tilde{\varphi}_{s,r}^{\Gamma^*}.$$

Integrating and letting the left hand side be zero, the prediction weights satisfy

$$\sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} a_{s,r}^{\Gamma^*} = 1.$$

Then, adapting the suggestion of Jansen et al. (2009), we construct the prediction weights by means of the normalised inverse distances,

$$a_{s,r}^{\Gamma^*} = \frac{1/\text{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_s^*}^*)}{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} 1/\text{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)}. \tag{17}$$

The weight construction above also has the appeal of naturally adapting to the local sampling regime, with distant neighbourhoods contributing less to prediction than nearby ones. We also consider a simple moving average prediction, with prediction weight as

$$a_{s,r}^{\Gamma^*} = \frac{1}{|\mathcal{N}_{k_r,r}^{\mathcal{V}^*}|}. \tag{18}$$

- **Update:** The update step is performed for the integrals and scaling coefficients associated with the neighbourhood $\{s : v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}\}$,

$$\begin{aligned} I_{s,r-1}^{\Gamma^*} &= I_{s,r}^{\Gamma^*} + a_{s,r}^{\Gamma^*} I_{k_r,r}^{\Gamma^*}; \\ c_{s,r-1}^{\Gamma^*} &= c_{s,r}^{\Gamma^*} + b_{s,r}^{\Gamma^*} d_{k_r}^{\Gamma^*}, \end{aligned} \tag{19}$$

with the update filter $\{b_{s,r}^{\Gamma*}\}$ obtained by the minimum norm solution as suggested in Jansen et al. (2009). Namely,

$$b_{s,r}^{\Gamma^*} = \frac{I_{s,r-1}^{\Gamma^*} I_{k_r,r}^{\Gamma^*}}{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} (I_{t,r-1}^{\Gamma^*})^2}, \quad \text{for } s : v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}. \tag{20}$$

These update coefficients along with the condition $\sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} a_{s,r}^{\Gamma^*} = 1$, guarantee the stability of the transform, see Jansen and Oonincx (2005) and Jansen et al. (2009) for more details.

- **Relink:** A further necessity through the lifting steps is to relink the graph structure, since the removal of a vertex (and of its associated edges)

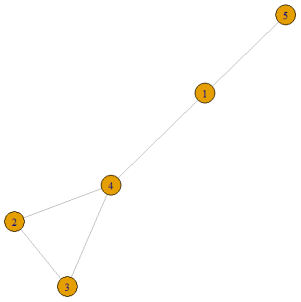

Fig. 2: An undirected network structure 'fiveNet' with five nodes from Knight et al. (2020).

might disconnect the graph structure. For a tree graph, it is easy to see that the tree will be disconnected by removing a vertex which is not on the boundary (the boundary of a graph consists of those vertices with only one neighbouring vertex). However, for a non-tree graph (graphs satisfying $|\mathcal{E}| \geq |\mathcal{V}|$), the remaining subgraph after removing a vertex and associated edges might still be connected. For example, Figure 2 is a toy network from Knight et al. (2020). Its graph structure is not that of a tree, and its structure will result in two separate components if we remove the first vertex and its associated edges, but the remaining graph will still be connected if we remove the second vertex and its associated edges.

Suppose we are at stage-r, then the relinkage part of the algorithm is carried out as follows.

1. Remove $v^*_{k_r}$ and all edges $\{e^*_l \,|\, v^*_{k_r} \in e^*_l\}$.
2. Test the connectivity of the subgraph consisting of $\mathcal{N}^{\mathcal{V}^*}_{k_r,r}$ as vertices. If connected, then the relinkage will not be performed.
3. If the subgraphs are not connected, then find the minimum spanning tree and embed this spanning tree into the existing graph structure.

Once we complete the relinkage, the neighbourhood structure will be updated to be $\mathcal{N}^{\mathcal{V}^*}_{k,r-1}$ for all $k \in \{1,...,m\}\backslash\{k_m,...,k_r\}$. We denote the new graph structure at stage-r as $G^*_r = (\mathcal{V}^*_r, \mathcal{E}^*_r)$, and the one after relinkage as $G^*_{r-1} = (\mathcal{V}^*_{r-1}, \mathcal{E}^*_{r-1})$, where $\mathcal{V}^*_r = \mathcal{V}^*_{r-1} \cup \{v^*_{k_r}\}$.

- **Iterate:** We iterate the split-predict-update steps discussed above to obtain a sequence of detail coefficients $\{d^{\Gamma^*}_{k_m},...,d^{\Gamma^*}_{k_{\tau+1}}\}$, where τ is the stopping time indicating the number of new vertices that will not be removed. In our work, based on thorough numerical investigation and as recommended in the literature for graph domain lifting but also for one- and two-dimensional cases both with real- and complex-valued filters (Nunes et al., 2006; Knight and Nunes, 2019), we retain $\tau = 2$ scaling coefficients. Mahadevan (2010) gives a detailed investigation on graph LOCAAT stopping times.

Inverse LG-LOCAAT. A lifting scheme such as the one discussed above is a linear transform and guarantees a perfect reconstruction (Sweldens, 1998). Thus, the inverse transform is carried out by *undoing* the lifting steps in equations (16) and (19), from stage-$(r-1)$ to stage-r,

$$c^{\Gamma^*}_{s,r} = c^{\Gamma^*}_{s,r-1} - b^{\Gamma^*}_{s,r} d^{\Gamma^*}_{k_r}, \quad \text{for } s : v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}, \tag{21}$$

$$c^{\Gamma^*}_{k_r,r} = d^{\Gamma^*}_{k_r} + \sum_{s:v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} c^{\Gamma^*}_{s,r}. \tag{22}$$

Function expansion. Through the iterations of the LG-LOCAAT steps, after stage-r we have

$$g^{\Gamma^*}(\mathbf{p}^*) = \sum_{s\in\mathcal{S}_{r-1}} c^{\Gamma^*}_{s,r-1}\varphi^{\Gamma^*}_{s,r-1}(\mathbf{p}^*) + \sum_{l\in\mathcal{D}_{r-1}} d^{\Gamma^*}_l \psi^{\Gamma^*}_l(\mathbf{p}^*), \tag{23}$$

where $\mathcal{D}_{r-1} = \{k_m,...,k_r\}$ are the wavelet coefficient indices, and $\mathcal{S}_{r-1} = \{1,...,m\}\backslash\mathcal{D}_{r-1}$ are the scaling indices.

Computational viewpoint. A lifting coefficient array has to be stored after every stage

to allow for the inverse transform to be carried out. For a tree structure at stage-$(r-1)$, we start with G^*_{r-1} and the set of edges $\{e^*_k = \{v^*_i, v^*_j\} \,|\, e^*_k \in \mathcal{E}^*_{r-1} \text{ and } v^*_i, v^*_j \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}\}$, then add the vertex $v^*_{k_r}$ and connect it to all vertices in $\mathcal{N}^{\mathcal{V}^*}_{k_r,r}$, thus obtaining G^*_r. For non-tree cases, it is possible that we do not disconnect some of the edges in the graph G^*_{r-1} as discussed in relinkage. Therefore, we have to preserve more information than for the lifting array in Jansen et al. (2009), with components

$$k_r \quad \mathcal{N}^{\mathcal{V}^*}_{k_r,r} \quad S^*_r \quad \underline{a}^{\Gamma^*}_r \quad \underline{b}^{\Gamma^*}_r,$$

where S^*_r is the set consists of all s such that $v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$, and $\underline{a}^*_r$, $\underline{b}^*_r$ are the prediction and update filters. In addition, for stage-r, all pairs (s, s'), where $s \neq s'$ and $s, s' \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$, such that $\{v^*_s, v^*_{s'}\} \in \mathcal{E}^*_r$ are added.

Theoretical considerations for the LG-LOCAAT construction appear in Appendix A, covering its sparsity, stability and scale definition.

4 Simulation study

In this section we investigate the behaviour of our proposed LG-LOCAAT in the context of a non-parametric regression problem carried out over the edge set of a network. We model the edge observations as

$$\begin{aligned} f^{\mathcal{E}}_k &= g^{\mathcal{E}}(e_k) + \epsilon_k, \\ &= g^{\mathcal{E}}_k + \epsilon_k. \end{aligned} \tag{24}$$

where $g^{\mathcal{E}}$ is the true, unknown function defined on the edge set $\mathcal{E} = \{e_k\}^m_{k=1}$, and $\{\epsilon_k\}^m_{k=1}$ is iid noise, assumed to follow a normal distribution $N(0, \sigma^2)$. Thus, $\underline{f}^{\mathcal{E}} = \{f^{\mathcal{E}}_k\}^m_{k=1}$ is the set of observation values on the edges of our graph G that are corrupted by noise, and our aim is to obtain an estimator $\hat{g}^{\mathcal{E}}$ of the true (unknown) function $g^{\mathcal{E}}$ evaluated at the observed edges.

The simulation study will assess three aspects indicative of the LG-LOCAAT's performance: stability (via the condition number), compression ability (via sparsity plots), and denoising performance (via the average mean squared error and associated estimator bias and variance).

4.1 Denoising strategy

Recall that the line graph transform allows us to represent $g^{\mathcal{E}}(e_k) = g^{\mathcal{V}^*}(v^*_k)$ (and similarly for the observations $\{f^{\mathcal{E}}_k\}^m_{k=1}$). Thus, equation (24) can be rewritten as

$$\begin{aligned} f^{\mathcal{V}^*}_k &= g^{\mathcal{V}^*}(v^*_k) + \epsilon_k \\ &= g^{\mathcal{V}^*}_k + \epsilon_k, \end{aligned}$$

and we perform the LG-LOCAAT decomposition for the observations $\underline{f}^{\mathcal{V}^*}$, in order to obtain a sequence of detail coefficients $\underline{d}^*$. Next, as introduced in Section 2.2, the detail coefficients are separated into artificial levels by taking a quantile split of the lifting continuous scales (integral values), see Jansen et al. (2009) and Nunes et al. (2006). The artificial level split ensures that the algorithm is capable to represent both the denser and the sparser sampled ares, in the zoom-in/zoom-out fashion typical of classical wavelet methods. Since in practice the noise has an unknown variance which has to be first estimated before applying thresholding, we follow Donoho and Johnstone (1994) and estimate σ by the median absolute deviation (MAD) of the detail coefficients belonging to the finest artificial level.

Exploiting the (true) wavelet coefficient sparsity, e.g. Sections 4.3.2 and C.2, wavelet thresholding will be performed on the detail coefficients. Following the finding from Mahadevan (2010) that thresholding around 80-90% detail coefficients appears to be an optimal choice, also verified by simulation results in our context here, throughout this work we perform wavelet thresholding for

all detail coefficients except for those allocated at the two coarsest artificial levels. We employ empirical Bayes thresholding, where the non-zero part of the prior density is modelled as the 'quasi-Cauchy' distribution from Johnstone and Silverman (2004), already proven to have good performance for LOCAAT-based approaches, see Nunes et al. (2006) and Jansen et al. (2009) for the one-dimensional and graph-contexts, respectively.

A set of estimated coefficients, $\hat{\underline{d}}^*$, will be obtained following thresholding and we then perform the inverse LG-LOCAAT transform in order to obtain an estimate of the true, unknown function. We denote the corresponding estimates as $\hat{g}_k^{\mathcal{V}^*}$ for all $k \in \{1, ..., m\}$ and we recall their equivalence to those in the original graph domain, such that $\hat{g}_k^{\mathcal{E}} = \hat{g}_k^{\mathcal{V}^*}$.

We investigate the effect of three different noise magnitudes, as measured by the signal-to-noise ratio (SNR = 3, 5, and 7). This ratio is given by $\text{SNR} = \sqrt{\text{var}(g^{\mathcal{E}})}/\sigma$, where $\text{var}(g^{\mathcal{E}})$ is the variance of the simulated true function, and σ is the standard deviation of the noise. For a clear comparison, the true function will be normalised so that $\text{var}(g^{\mathcal{E}}) = 1$.

We sample $q = 1, ..., Q$ different graph structures, $G^{(q)}$. For each of them, we simulate $r = 1, ..., R$ different noise sequences, corresponding to a certain noise level, as discussed above. Following the estimation procedure, we calculate the average mean squared error (AMSE) defined as

$$\text{AMSE} = (QRm)^{-1} \sum_{q=1}^{Q} \sum_{r=1}^{R} \sum_{k=1}^{m} \left(\hat{g}_{k,q,r}^{\mathcal{E}} - g_{k,q}^{\mathcal{E}}\right)^2,$$

where $g_{k,q}^{\mathcal{E}}$ is the true edge function value corresponding to the k-th 'new' vertex on the line graph $G^{*(q)}$, while $\hat{g}_{k,q,r}^{\mathcal{E}}$ is the estimate of $g_{k,q}^{\mathcal{E}}$ when the true function is corrupted by the r-th noise sequence.

We also investigate the variance and squared bias associated with our methods, where

$$\text{Var} = (QRm)^{-1} \sum_{q=1}^{Q} \sum_{k=1}^{m} \sum_{r=1}^{R} \left(\hat{g}_{k,q,r}^{\mathcal{E}} - \bar{g}_{k,q}^{\mathcal{E}}\right)^2,$$

where $\bar{g}_{k,q} = \frac{1}{R} \sum_{r=1}^{R} \hat{g}_{k,q,r}^{\mathcal{E}}$ and the squared bias is calculated as

$$\text{Bias}^2 = (Qm)^{-1} \sum_{q=1}^{Q} \sum_{k=1}^{m} \left(\bar{g}_{k,q}^{\mathcal{E}} - g_{k,q}^{\mathcal{E}}\right)^2.$$

The true function $g^{\mathcal{E}}$ is generated by the test functions as described next.

4.2 Test Functions

The set of functions from Jansen et al. (2009) along with the g_1 function introduced in Mahadevan (2010) will be used in our simulation study in order to capture the variety of traits that may occur in signals recorded over the edges of networks pertaining to fields from hydrology and transportation to computer networks. Figure 3 illustrates that 'Blocks' is piecewise continuous with several discontinuities; 'Doppler' is a smooth function; 'Bumps' displays several spikes; 'Heavisine' is a smooth function but with high variability; '`mfc`' presents two discontinuous sections, while the g_1 has a similar but finer structure.

4.2.1 Sampling Network Structure

Since the test functions are defined over the square $[0, 1] \times [0, 1]$ (the formulae can be found in Mahadevan (2010)), we sample this space by means of a network structure. We first sample n points $\{(x_i, y_i)\}_{i=1}^n$, where $x_i, y_i \sim \text{Unif}(0, 1)$. These points are fixed as the graph vertices $\{v_i = (x_i, y_i)\}_{i=1}^n$ of the network G. The graph edges are obtained by the minimum spanning tree, which gives us a set of m ($m = n - 1$) connections between vertices. We let these be the set of edges

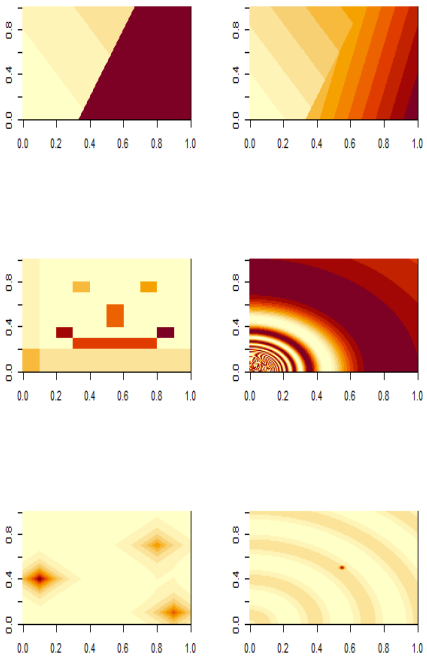

Fig. 3: Test functions heat maps. From left to right on *top row*: g_1, maartenfunc (mfc); *middle row*: Blocks, Doppler; *bottom row*: Bumps, Heavisine.

for the graph G, with the length of each edge given by the Euclidean distance between the two vertices associated with this edge.

An example appears in Figure 4, displaying the network vertices and edges superimposed with the Blocks function, evaluated as next described and visualised using a Voronoi polygon tessellation centred at the edge midpoints.

4.2.2 Embedding the Function Values

Within our study, we evaluate the function values over the network edges using two methods. The first method is to simply select the value that corresponds to the coordinate at the midpoint of each edge. Let us assume we have a

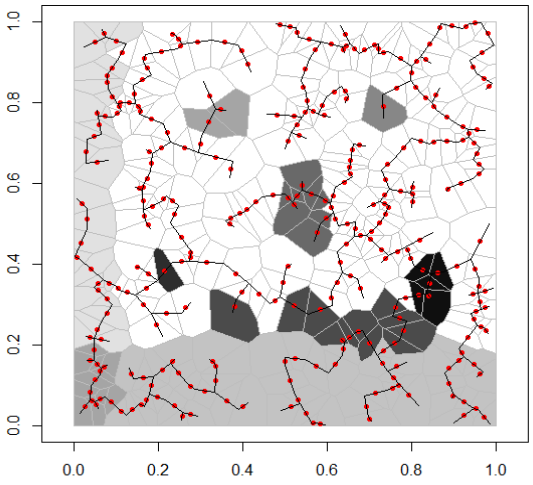

Fig. 4: Pointwise Blocks function. Each Voronoi polygon represents the function value at the corresponding new vertex (original edge), represented by the red points.

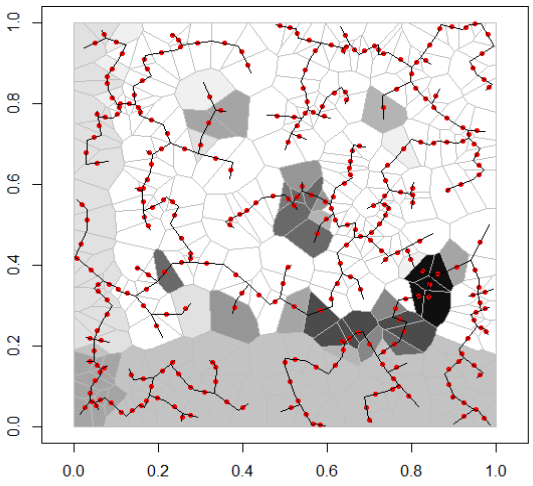

Fig. 5: Edge averaging Blocks function. Each Voronoi polygon represents the function value at the corresponding new vertex (original edge), represented by the red points.

function g^{test} from the previously mentioned collection of test functions. Consequently, for an edge $e_k = \{v_i, v_j\}$, where $v_i = (x_i, y_i)$ and $v_j = (x_j, y_j)$, the corresponding 'true' observation value will be

$$g_k^{\mathcal{E}} = g_k^{\mathcal{V}^*} := g^{\text{test}}\left(\frac{x_i + x_j}{2}, \frac{y_i + y_j}{2}\right). \tag{25}$$

Nonetheless, this definition still hinges on a vertex-wise perspective and we refer to these as 'pointwise' functions. We additionally explore an alternative method for designing the function values, which incorporates the network geometry. Inspired by the cell averaging of Donoho (1997), we view the edges as line segments and for $e_k = \{v_i, v_j\}$, we define the function value as

$$g_k^{\mathcal{E}} = g_k^{\mathcal{V}^*} := \\ = \frac{1}{N}\sum_{h=0}^{N-1} g^{\text{test}}\left(x_i + \frac{x_j - x_i}{N-1}h, y_i + \frac{y_j - y_i}{N-1}h\right). \tag{26}$$

We refer to these as 'edge averaging' functions. Here we let $N = 100$. Figures 4 and 5 show the 'pointwise' and 'edge averaging' Blocks functions, respectively. Note that when compared with the pointwise function, the edge averaging is smoother when the edges cross blocks with different values.

4.3 Results

In this section, we will provide numerical results for the proposed LG-LOCAAT algorithm which gives rise to several variants determined by specific combinations of initial scaling function integral definition and prediction weights, as well as the availability of coordinate information or only of the path length. The acronyms we use to identify the approaches we explore, as well as their brief descriptions, appear in Table 1.

4.3.1 Stability

As the lifting scheme is a linear transform, it can be represented using matrix multiplication, $\underline{d} = \tilde{R}\underline{g}$, where $\underline{g}$ is the observation sequence, $\tilde{R}$ denotes the forward matrix generated by lifting scheme (see Nunes et al. (2006) for details on its LOCAAT construction), and $\underline{d}$ is the detail coefficient sequence. The condition number of the lifting matrix will be used to assess the stability of the transform and can be computed as $\kappa(\tilde{R}) = \rho_1/\rho_m$, where $\{\rho_i\}_{i=1}^m$ is the set of corresponding singular values ordered according to their magnitude, such that ρ_1 is the maximum singular value and ρ_m is the minimum one. The condition number satisfies $\kappa(\tilde{R}) \geq 1$ and the lower its value, the more stable the transform (Higham, 2002) is.

Tables C1 and C2 illustrate that all algorithms are characterised by similar condition numbers, with a preference for initial integrals as a sequence of ones and moving average prediction weights.

4.3.2 Sparsity

The tool we use to assess the performance of compression is the sparsity plot, constructed as follows. First, we perform the LG-LOCAAT transform on the sequence of true values $\{g_k^{\mathcal{V}^*}\}_{k=1}^m$. This will yield a corresponding vector which contains two scaling coefficients and $(m-2)$ detail coefficients. We begin with just two scaling coefficients and assume all detail coefficients are zero, then perform the inverse transform for this vector. In general, obtain an estimator $\hat{g}^{\mathcal{V}^*}(t)$ for the true function $g^{\mathcal{V}^*}$, where $(t-1)$ gives the number of detail coefficients used in reconstruction, such that $\hat{g}^{\mathcal{V}^*}(1)$ is the reconstruction with only two scaling coefficients, while $\hat{g}^{\mathcal{V}^*}(2)$ means the reconstruction with two scaling coefficients and one detail coefficient with the largest absolute value, and so on. For any step, we will introduce the detail coefficient with the largest absolute value

Acronym	Proposed LG-LOCAAT variant
LG-Sid-c	S: sum of distances as integral (equation (13)); id: inverse distance prediction (equation (17)); c: coordinate information available.
LG-Aid-c	A: average distance as integral (equation (14)); id: inverse distance prediction (equation (17)); c: coordinate information available.
LG-Did-c	D: a sequence of ones as integrals (equation (15)); id: inverse distance prediction (equation (17)); c: coordinate information available.
LG-Snw-c	S: sum of distances as integral (equation (13)); nw: moving average prediction (equation (18)); c: coordinate information available.
LG-Anw-c	A: average distance as integral (equation (14)); nw: moving average prediction (equation (18)); c: coordinate information available.
LG-Dnw-c	D: a sequence of ones as integrals (equation (15)); nw: moving average prediction (equation (18)); c: coordinate information available.
LG-Sid-p	S: sum of distances as integral (equation (13)); id: inverse distance prediction (equation (17)); p: path length available.
LG-Aid-p	A: average distance as integral (equation (14)); id: inverse distance prediction (equation (17)); p: path length available.
LG-Did-p	D: a sequence of ones as integrals (equation (15)); id: inverse distance prediction (equation (17)); p: path length available.
LG-Snw-p	S: sum of distances as integral (equation (13)); nw: moving average prediction (equation (18)); p: path length available.
LG-Anw-p	A: average distance as integral (equation (14)); nw: moving average prediction (equation (18)); p: path length available.
LG-Dnw-p	D: a sequence of ones as integrals (equation (15)); nw: moving average prediction (equation (18)); p: path length available.

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

into the vector for reconstruction. We thus obtain a different estimate at each step, until all detail coefficients have been used. For the sparsity plot, the x-axis is the 't'-argument and the value on the y-axis is the integrated squared error (ISE), defined as

$$\mathrm{ISE}(t) = Q^{-1} \sum_{q=1}^{Q} \sum_{k=1}^{m} \left(\hat{g}_k^{\mathcal{V}^{*(q)}}(t) - g_k^{\mathcal{V}^{*(q)}} \right)^2,$$

where $g_k^{\mathcal{V}^{*(q)}}$ and $\hat{g}_k^{\mathcal{V}^{*(q)}}$ denote the true observation at new vertex v_k^* and its reconstruction uses $(t-1)$ detail coefficients for the q-th network. If the ISE decays to zero fast (if for small t, the ISE is already close to zero), then the algorithm leads to a sparse decomposition for the target function.

Results for Pointwise Functions. Figure C3 shows the sparsity results across the (pointwise) test functions, for schemes based on the coordinate information. Note that the compression ability of the algorithm on g_1 and `mfc` surpasses that on any other test function. Bumps and Doppler functions display similar sparsity results, while for Blocks the results are inferior. The compression

of Heavisine is the weakest compared with other functions. Reassuringly, there is no significant difference among the choices of integral values and prediction weights. Figure C4 shows the results for the same functions while using path length instead of the coordinates. For data compression, there is no significant difference between using path length and coordinate information.

Results for Edge Averaging Functions. Figure C5 shows the sparsity results for different edge averaging functions. The compression ability follows the similar patterns to pointwise functions. Along with Figure C6, again there is no evidence of a significant difference between using path length and coordinate information. Both sets of plots indicate that different choices of integral values and prediction weights will not significantly affect the algorithm's compression ability.

4.3.3 Denoising Performance

In this section, we investigate the behaviour of LG-LOCAAT in the context of the network edge non-parametric regression problem outlined in Section 4.1 using $R = 100$ noise sequences over $Q = 50$ graph structures each with $m = 99$ edges.

Results for Pointwise Functions. In terms of AMSE, Table 2 illustrates that using the average distance as the integral value surpasses the choice of sequence of ones or sum integral choices. Methods 'Did' and 'Dnw' perform well for most of the function except Blocks and Heavisine, while 'Sid' and 'Snw' never appear to be the optimal choice unless the function is the Blocks (piecewise constant). From Table C3, we note that the integral choice 'A' (average distance) provides the best results for variance control, while the choice 'S' (sum of distances) provides similar and competitive results, too. Performing the algorithm with the integral choice 'D' (sequence of ones) gives a relatively high variance compared with the other two choices. The reason for this might be the dual wavelet functions have fewer overlaps than any other scaling function construction choices (recall that the integral choice 'D' allows us to capture the information more uniformly on the graph structure). For the bias results in Table C4, the integral choice 'D' surpasses the other methods for most of the functions. The choice 'A' is competitive for the Heavisine function, which contains high frequency components when compared with other functions. The choice 'S' introduces high bias for the smoother functions (Doppler, Heavisine, `mfc`). Hence, when using coordinate information, overall 'LG-Aid-c' appears to be a balanced choice that delivers competitive results irrespective of signal smoothness and noise contamination level. Although 'LG-Did-c' does not give as good results as 'LG-Aid-c' in terms of AMSE (especially for Blocks and Heavisine) and variance, it is still worth emphasising on since it yields a low bias. Let us next investigate the impact of not accessing the coordinate information.

Performing the algorithm by using the path distance instead of coordinate information, the AMSE results are remarkably comparable to those when coordinate information is available, see Table 3. The variance and bias patterns in Tables C5 and C6 appear similar to previous methods. However, should coordinate information be available, this may be the better choice. This may be justified by the design of the function values based on a two-dimensional Euclidean space and it will be essential to inspect the algorithm performance for the edge averaging functions (see next section).

Additionally, the average distance for integral and the inverse distance prediction ('LG-Aid-p') also arise as a strong choice in this context, with the 'Did' choice a close competitor in terms of bias control, and 'Sid' a close match particularly for variance results. The performance of 'LG-Sid-p'

is very similar to 'LG-Aid-p' except for Heavisine. So using 'average distances' as integral is advantageous when dealing with high-frequency information, as opposed to using 'sum of distances' as integral. However, we note that the choice of average distances as integral values lacks theoretical support.

Results for Edge Averaging Functions. Tables 4 and 5 illustrate that when employing our algorithm on edge averaging functions, the 'Aid' algorithm is the optimal choice in terms of the AMSE when the coordinate information is known, while 'Did' performs better if only path lengths are known. However, 'Aid' surpasses the other methods for high frequency Heavisine function. Choosing a sequence of ones as the starting integrals decreases the bias for all functions, and the improvements are more significant when we use the path length instead of the coordinate information (Tables C8 and C10). Again, 'Aid' is the optimal choice in terms of variance control (Tables C7 and C9). The methods with coordinate information still slightly surpass the corresponding results by the path distance, except for mfc. According to AMSE, the performances of 'Did' and 'Aid' are close except when the underlying function has high frequency (e.g., Heavisine).

Overall, we recommend the use of coordinate information when available, as implemented in the methods, LG-Aid-c and LG-Did-c, if the underlying function are spatial coordinate dependent. Should such coordinate information not be available, LG-Aid-p and LG-Did-p are also very competitive throughout the board.

The median running time for the simulations above (50 networks with 100 noise replications) is 00:49:30 (hours:minutes:seconds), conducted using the York Viking high-performance computing facility (https://vikingdocs.york.ac.uk/).

5 Hydrological data analysis

In this section we tackle the dissolved oxygen (DO) dataset that motivated this work, where DO measures the amount of oxygen in water and acts as an indicator of water quality. In order to assess the behaviour of our proposed algorithms on hydrological data, we take an additional step through first analysing a piecewise constant function mimicking river flow, as introduced by Park and Oh (2022) on a simulated tree network from Gallacher et al. (2017). This simulated flow data can be visualised in Figure 6 and is also investigated as a real data-like example by Park and Oh (2022), who evaluate their proposed method on it.

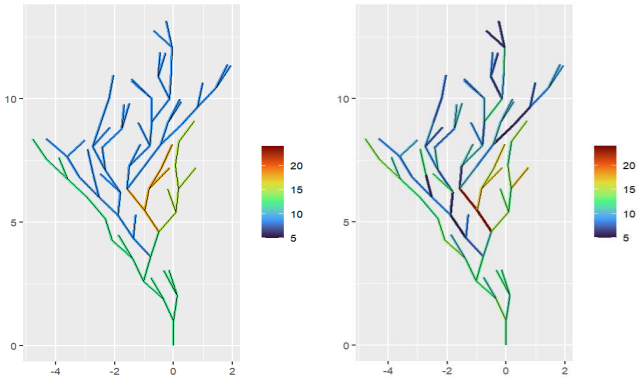

Fig. 6: **Left:** The simulated river flow data, the network structure is introduced in Gallacher et al. (2017) and the test function construction is from Park and Oh (2022). **Right:** Noise corrupted flow data using $\epsilon \sim N(0, 4)$.

5.1 Flow-based Function Denoising

The river network contains 80 vertices and 79 edges, and its edge set is separated into seven different clusters. The flow function is constructed as follows (Park and Oh, 2022). The function values for every stream are initially set as 9, then a cluster is randomly picked and the function values

AMSE$\times 10^3$ (sd$\times 10^3$)	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
			SNR=3			
LG-Sid-c	65 (19)	114 (44)	109 (39)	92 (26)	272 (99)	52 (12)
LG-Aid-c	**62** (18)	**113** (48)	**92** (29)	**79** (19)	**198** (66)	46 (10)
LG-Did-c	64 (18)	120 (46)	93 (31)	80 (20)	235 (81)	**45** (11)
LG-Snw-c	66 (19)	117 (47)	109 (38)	97 (28)	282 (103)	53 (12)
LG-Anw-c	64 (18)	116 (48)	**92** (31)	81 (20)	205 (66)	46 (10)
LG-Dnw-c	67 (19)	126 (51)	98 (33)	83 (21)	261 (87)	47 (11)
			SNR=5			
LG-Sid-c	23 (7)	**41** (16)	44 (17)	45 (17)	206 (88)	26 (6)
LG-Aid-c	23 (7)	42 (18)	38 (13)	**38** (12)	**138** (48)	**22** (5)
LG-Did-c	23 (7)	46 (20)	**37** (13)	39 (12)	178 (76)	**22** (5)
LG-Snw-c	**22** (7)	42 (17)	44 (17)	47 (18)	217 (91)	26 (6)
LG-Anw-c	23 (8)	43 (19)	38 (14)	39 (12)	147 (48)	**22** (5)
LG-Dnw-c	24 (8)	48 (21)	39 (14)	41 (13)	208 (81)	23 (5)
			SNR=7			
LG-Sid-c	11 (3)	**21** (9)	24 (10)	28 (13)	184 (83)	17 (4)
LG-Aid-c	11 (3)	22 (10)	**21** (7)	**23** (8)	**120** (43)	**14** (3)
LG-Did-c	11 (3)	24 (11)	**21** (7)	24 (9)	161 (75)	**14** (3)
LG-Snw-c	**10** (3)	**21** (9)	25 (10)	29 (14)	195 (86)	18 (4)
LG-Anw-c	**10** (3)	22 (12)	22 (8)	24 (8)	130 (43)	**14** (3)
LG-Dnw-c	11 (3)	25 (11)	22 (8)	25 (9)	191 (79)	15 (3)

Table 2: AMSE for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the pointwise construction. We assume the coordinate information is available. The values in parentheses are the standard deviations ($\times 10^3$) of the AMSE results across the $Q \times R = 50 \times 100$ replications.

of every stream in this cluster will be randomly chosen from the values $\{12, 15, 18\}$. We continue this procedure until there are more than 30 edges containing a value larger than 9. Figure 6 (right panel) shows the noise corrupted flow data.

5.1.1 Nondecimated lifting transform

For completeness, we formally bring the nondecimation concept into the graph data setup. Knight and Nason (2009) introduced a nondecimated lifting transform (NLT) based on the LOCAAT framework from Jansen et al. (2009) where instead of using the shift operator, their proposal was to use different 'trajectories' of removal order.

In the graph context here, where we have for an unknown function $g^{\mathcal{E}} : \mathcal{E} \longrightarrow \mathbb{R}$ the set of noisy observations $\{f_k^{\mathcal{E}}\}_{k=1}^m$ on the graph edge set $\{e_k\}_{k=1}^m$, we propose to naturally follow the idea from Knight and Nason (2009) and generate P trajectories $\{T_p\}_{p=1}^P$, where $T_p = (e_{o_1}, ..., e_{o_m})$ and $(o_1, ..., o_m)$ is a permutation of the ordered sequence $(1, ..., m)$. For each trajectory, let us say T_p, a detail coefficient sequence $\underline{d}^{*,(p)}$ can be obtained by any of (as long as it is the same one) the proposed transforms. Then an estimator $\hat{g}^{\mathcal{E},(p)}$

AMSE$\times 10^3$ (sd$\times 10^3$)	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-p	67 (21)	**116** (45)	109 (39)	93 (25)	274 (96)	54 (12)
LG-Aid-p	**65** (20)	117 (48)	**94** (31)	84 (21)	**209** (73)	**47** (10)
LG-Did-p	**65** (19)	122 (49)	95 (31)	**83** (22)	253 (93)	**47** (11)
LG-Snw-p	68 (21)	118 (48)	110 (36)	96 (26)	288 (102)	54 (12)
LG-Anw-p	66 (20)	124 (55)	95 (31)	84 (22)	215 (76)	48 (10)
LG-Dnw-p	68 (20)	129 (54)	100 (33)	86 (22)	276 (93)	48 (11)
SNR=5						
LG-Sid-p	**23** (8)	**42** (17)	44 (17)	45 (16)	208 (83)	27 (7)
LG-Aid-p	**23** (8)	44 (19)	**38** (13)	**40** (13)	**152** (62)	**23** (5)
LG-Did-p	24 (8)	47 (21)	**38** (13)	**40** (13)	192 (86)	**23** (5)
LG-Snw-p	**23** (8)	**42** (19)	44 (16)	47 (17)	220 (87)	27 (6)
LG-Anw-p	24 (8)	46 (22)	39 (13)	**40** (13)	159 (61)	**23** (5)
LG-Dnw-p	25 (8)	49 (22)	40 (14)	42 (13)	219 (89)	24 (5)
SNR=7						
LG-Sid-p	**11** (3)	22 (10)	24 (10)	28 (12)	186 (78)	18 (5)
LG-Aid-p	**11** (3)	23 (11)	**21** (7)	**24** (9)	**135** (58)	**15** (3)
LG-Did-p	**11** (4)	25 (12)	**21** (7)	25 (9)	172 (83)	**15** (3)
LG-Snw-p	**11** (3)	**21** (10)	25 (10)	29 (13)	197 (80)	18 (5)
LG-Anw-p	**11** (3)	24 (13)	22 (8)	**24** (9)	142 (55)	**15** (3)
LG-Dnw-p	**11** (4)	26 (12)	22 (8)	26 (10)	201 (87)	16 (4)

Table 3: AMSE for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the pointwise construction. The path distance is used. The values in parentheses are the standard deviations ($\times 10^3$) of the AMSE results across the $Q \times R = 50 \times 100$ replications.

will be obtained corresponding to each trajectory T_p, and we propose to use the average estimator

$$\hat{\bar{g}}_k^{\mathcal{E}} = \frac{1}{P}\sum_{p=1}^{P} \hat{g}_k^{\mathcal{E},(p)},$$

for $k \in \{1, ..., m\}$. In the work from Knight and Nason (2009), a genetic algorithm (Lucasius and Kateman, 1993, 1994) is applied to generate 'well-behaved' trajectories, which are those likely to have low variations. In the context of applying the LOCAAT algorithm for river network applications, Park and Oh (2022) suggest to perform several permutations within each stream cluster for each trajectory. In this work, we assume that there is no prior information on the underlying function, thus, the trajectories are chosen by random permutations, but note that further improvements may be possible as suggested by Knight and Nason (2009) and Park and Oh (2022). Additionally, although not implemented here, we note that the bootstrap nature of the nondecimated algorithms may be used in turn to construct empirical confidence bands for the target function.

AMSE$\times 10^3$ (sd$\times 10^3$)	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-c	64 (19)	126 (47)	103 (32)	92 (25)	281 (96)	51 (12)
LG-Aid-c	**59** (16)	**120** (48)	89 (27)	**79** (18)	**206** (70)	45 (10)
LG-Did-c	**59** (16)	**120** (43)	**88** (26)	**79** (19)	228 (77)	**44** (10)
LG-Snw-c	65 (19)	128 (46)	104 (32)	96 (27)	292 (101)	52 (12)
LG-Anw-c	60 (16)	125 (46)	90 (28)	80 (19)	215 (70)	45 (10)
LG-Dnw-c	61 (16)	128 (46)	92 (28)	82 (20)	255 (83)	46 (11)
SNR=5						
LG-Sid-c	26 (9)	50 (22)	45 (17)	45 (16)	220 (86)	25 (6)
LG-Aid-c	**24** (8)	**49** (23)	39 (14)	**38** (11)	**147** (53)	**21** (4)
LG-Did-c	25 (7)	51 (21)	**38** (14)	**38** (12)	176 (71)	**21** (4)
LG-Snw-c	26 (9)	51 (22)	46 (18)	47 (17)	230 (90)	26 (6)
LG-Anw-c	25 (9)	51 (24)	40 (14)	39 (12)	158 (53)	22 (5)
LG-Dnw-c	26 (8)	54 (23)	40 (14)	40 (12)	204 (76)	22 (5)
SNR=7						
LG-Sid-c	**13** (4)	**27** (12)	26 (11)	28 (12)	199 (81)	17 (4)
LG-Aid-c	**13** (4)	**27** (13)	**22** (8)	**23** (8)	**129** (48)	14 (3)
LG-Did-c	**13** (4)	28 (13)	**22** (8)	24 (8)	160 (69)	**13** (3)
LG-Snw-c	**13** (4)	**27** (12)	26 (11)	29 (13)	211 (86)	18 (5)
LG-Anw-c	**13** (5)	**27** (15)	23 (9)	24 (8)	141 (47)	14 (3)
LG-Dnw-c	**13** (5)	30 (14)	23 (8)	25 (9)	190 (74)	15 (3)

Table 4: AMSE for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the edge-averaging construction. We assume the coordinate information is available. The values in parentheses are the standard deviations ($\times 10^3$) of the AMSE results across the $Q \times R = 50 \times 100$ replications.

5.1.2 Flow Denoising Results

We now investigate our proposed algorithms and their nondecimated versions on the noise contaminated flow dataset. Specifically, we employ the 'LG-Sid-p' and 'LG-Aid-p' algorithms with the path length instead of using the coordinate information, since the proposals in Park and Oh (2022) are also based on path length metric and therefore this ensures a fair comparison. Table 6 shows the results when using our algorithms (one trajectory, following the minimum integral paradigm) as well as their nondecimated versions (acronyms ending in 'nlt' in Table 6). For completeness, we also display the results of the methods from Park and Oh (2022) along with their reproduced results of the method from O'Donnell et al. (2014), as well as employing Gaussian Whittle-Matérn fields (Bolin et al., 2023a, 2024) as a competitor. The latter computations and flow data analysis visualisations, including the error plots in Figure 7, were carried out using the `MetricGraph` R package (Bolin et al., 2023b).

AMSE$\times 10^3$ (sd$\times 10^3$)	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-p	66 (20)	127 (47)	104 (32)	93 (25)	282 (95)	53 (12)
LG-Aid-p	62 (17)	123 (47)	91 (28)	83 (21)	**218 (76)**	**46 (10)**
LG-Did-p	**60** (17)	**122** (45)	**90** (27)	**82** (21)	247 (87)	**46** (11)
LG-Snw-p	66 (21)	129 (48)	104 (30)	96 (26)	296 (101)	53 (12)
LG-Anw-p	62 (17)	133 (51)	92 (28)	84 (22)	225 (77)	47 (10)
LG-Dnw-p	62 (17)	130 (49)	94 (28)	85 (22)	270 (89)	47 (11)
SNR=5						
LG-Sid-p	26 (9)	**51** (23)	45 (17)	46 (16)	222 (85)	26 (7)
LG-Aid-p	26 (9)	52 (24)	**39** (14)	40 (12)	**162 (64)**	**22 (5)**
LG-Did-p	**25** (8)	52 (22)	**39** (14)	**39 (12)**	190 (82)	**22 (5)**
LG-Snw-p	26 (9)	52 (23)	46 (17)	47 (17)	233 (88)	27 (6)
LG-Anw-p	26 (9)	55 (27)	41 (14)	40 (12)	170 (63)	23 (5)
LG-Dnw-p	26 (8)	55 (25)	41 (15)	41 (13)	217 (85)	23 (5)
SNR=7						
LG-Sid-p	**13** (5)	**27** (13)	26 (11)	28 (12)	202 (81)	18 (5)
LG-Aid-p	**13** (4)	28 (15)	**22** (8)	**24** (9)	**145** (60)	**14** (3)
LG-Did-p	**13** (4)	29 (14)	**22** (8)	**24** (9)	173 (78)	**14** (3)
LG-Snw-p	**13** (5)	**27** (13)	26 (11)	29 (12)	212 (81)	18 (5)
LG-Anw-p	14 (5)	29 (16)	23 (8)	25 (9)	154 (57)	15 (3)
LG-Dnw-p	**13** (5)	31 (15)	23 (9)	26 (9)	201 (83)	16 (3)

Table 5: AMSE for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the edge-averaging construction. The path distance is used. The values in parentheses are the standard deviations ($\times 10^3$) of the AMSE results across the $Q \times R = 50 \times 100$ replications.

We observe that for the simulated flow data with the proposed trajectory choice, 'LG-Sid-p' results surpass all other (decimated) results when the noise level is not very high ($\sigma = 1$ and $\sigma = 1.5$). The Gaussian Whittle–Matérn fields method performs better than other (one trajectory) methods. Notably, with only one choice of trajectory, we obtain better results than the nondecimated results (50 trajectories) from Park and Oh (2022), while NLT has been shown in the literature to drastically improve denoising results (Knight and Nason, 2009).

The nondecimated line graph algorithms work better than the other choices, especially 'LG-Aid-p-nlt', which gives consistently most competitive results, regardless of the noise level, as (very closely) does 'LG-Sid-p-nlt'.

We also note here the usefulness of introducing the random trajectory choice via nondecimation through the use of averaged estimators. In particular, exploring 30 trajectories has a significant impact on lowering the denoising errors, especially for lower signal-to-noise ratios. For the highest reported noise level, the percentage improvement by introducing several trajectories is around 23.8%

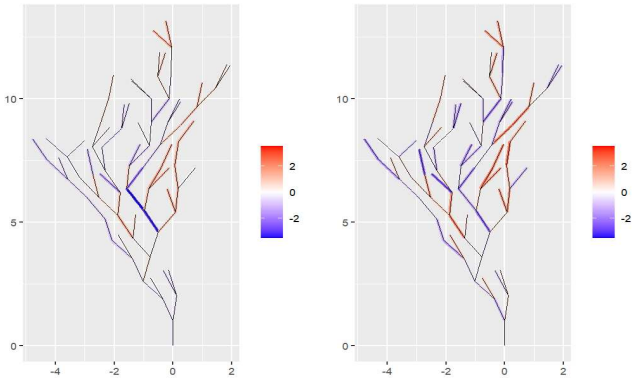

Fig. 7: **Left:** Error plot for river flow data denoised by 'LG-Aid-p-nlt' with 30 trajectories. **Right:** Error plot for river flow data denoised by Whittle-Matérn fields (Bolin et al., 2023a, 2024).

for 'LG-Aid-p-nlt'. The percentage improvement decreases with the decreasing noise level, but it is still substantial for our methods, ranging from around 21.4% for 'LG-Aid-p-nlt' to the lowest improvement is for 'LG-Sid-p-nlt' (although this notably has a more competitive starting value for one trajectory), nevertheless still sizeable at 14.1%, 16.7%, and 18.2% corresponding to the reported noise level ordering in Table 6.

Figure D7 shows the denoised signal by 'LG-Aid-p-nlt', and we note that our algorithm is able to capture the true underlying function structure.

Additionally, we also compare the computational cost (when using a standard specification personal laptop) of the three strong competitors, 'LG-Sid-p', 'LG-Aid-p-nlt (30)' and Gaussian Whittle-Matérn fields, by reporting the execution time of the simulations in Table 6. The elapsed time of the Gaussian Whittle-Matérn fields method is 50.1 minutes, with a 47.6 minute user time and a 1.4 minute system time. For 'LG-Sid-p', the elapsed time is 1.5 minutes with a 1.4 minute user time and a 0.0 minute system time; while the elapsed time of 'LG-Aid-p-nlt (30)' is 60.5 minutes, with a 57.4 minute user time and a 1.7 minute system time. Hence our proposed algorithms are competitive in both computation and denoising performance. Note that one may choose fewer trajectories, e.g., 15 runs for the NLT, in order to alter the balance between denoising performance and computational cost.

5.2 Real Data Analysis

In this section, we carry out our proposed algorithms 'LG-Aid-nlt' and 'LG-Did-nlt' on the DO dataset. Recall we argued that it is natural to consider data collected from stations as edge-based observations, since each station is connected to a certain river body. In order to obtain the whole network structure, we also need to define the vertices. A natural choice would be to consider river conjunctions, the sources and mouths of rivers as the vertices, with each river being recognised as an edge. However, through this construction the England river network would be a tree structure with a large number of vertices, while only 60 observations are available. Even assuming the whole network structure has been constructed, some information would still be difficult to obtain, for example, the length of an edge (river) since each river is a curve instead of a line segment.

Our proposed line graph-based construction crucially enables us to bypass these problems and allows us the framework in which to model the stations as the line graph vertices (Figure 8 illustrates a local area enhanced with line graph connectivity). Here, we propose to consider their associated latitude and longitude as their coordinates.

Remark (Neighbourhood Structure). In the work from Park and Oh (2022), the neighbourhood selection is based on the concept of 'flow-connectedness' introduced by Hoef et al. (2006). Under this concept, if the intersection of upstreams of two stations is a non-empty set, then they are defined as neighbours. In our work, we conjecture that river quality related indices are

RMSE (Std. error)	80 obs ($\sigma = 1$)	80 obs ($\sigma = 1.5$)	80 obs ($\sigma = 2$)
LG-Sid-p	0.6637 (0.0937)	0.9770 (0.1428)	1.2651 (0.1844)
LG-Sid-p-nlt (30)	0.5704 (0.0779)	0.8132 (0.1127)	1.0346 (0.1433)
LG-Aid-p	0.7183 (0.1139)	1.0484 (0.1480)	1.3429 (0.1932)
LG-Aid-p-nlt (30)	**0.5645 (0.0786)**	**0.8050 (0.1144)**	**1.0231 (0.1456)**
Proposed (Median) from Park and Oh (2022)	0.7265 (0.1212)	1.0249 (0.1510)	1.2818 (0.2599)
Proposed (Hard) from Park and Oh (2022)	0.7396 (0.1317)	1.0666 (0.1678)	1.3705 (0.2495)
Proposed (Median, nlt) from Park and Oh (2022)	0.7162 (0.1219)	1.0106 (0.1678)	1.2816 (0.2712)
O'Donnell	1.2698 (0.1251)	1.3815 (0.1727)	1.5421 (0.2017)
Whittle–Matérn fields	0.7699 (0.1201)	1.0040 (0.1911)	1.1882 (0.2655)
Whittle–Matérn fields with spatial coordinates	0.7804 (0.1203)	1.0193 (0.1936)	1.2438 (0.2986)

Table 6: Average root mean squared error (and associated standard error) for different methods performed on simulated flow data.

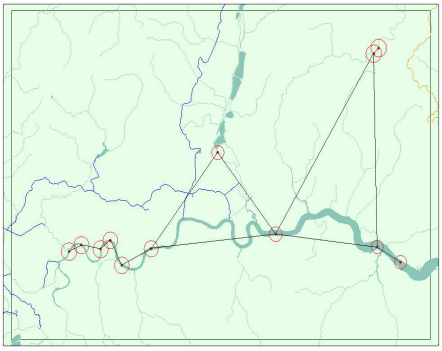

Fig. 8: DO visualisation for London and nearby area enhanced with line graph vertices (red circles for observation stations) and edges (black lines).

highly dependent on both river network structure and on localised human and animal behaviour. Hence, defining the neighbourhood structure as in equation (7) and performing the LG-LOCAAT algorithm with coordinate information is useful since it enables us to capture these traits.

DO Dataset Results. Guided by the results from previous simulation studies, we carry out our analysis by the denoising strategy described in Sections 4.1 and 5.1.1 using the nondecimated versions of our proposed 'LG-Aid-c' and 'LG-Did-c' for obtaining the detail coefficients. Since increasing the number of trajectories in nondecimated lifting increases the denoising performance, we perform a 100-trajectory nondecimated lifting on the DO dataset.

Figure 9 shows the denoised version of the DO data shown in Figure 1 (left), as well as the residuals following our estimation (right). We observe that our methods tend to smooth the observations based on the network structure, see the red circle chain from Wales to the southeast of England, and the values in northern England. Also note that the DO values tend to be large around the countryside and areas close to the sea thus indicating good water quality, with low values around the London and Manchester areas, following the intuition that big cities have a negative impact on the water quality.

In order to illustrate the impact of the underlying network on the quality of the estimate, we also carry out the data analysis for data collected at two different time snapshots. The residuals in Figure 10 for the data collected on 10/May/2024 appear to closely follow the normal distribution. However, the residual Q-Q plots in Figure 11 for the data collected on 05/Jun/2024 are less well behaved, especially for the 'LG-Did' algorithm that uses a constant integral initialisation. This highlights the network impact on the quality of the

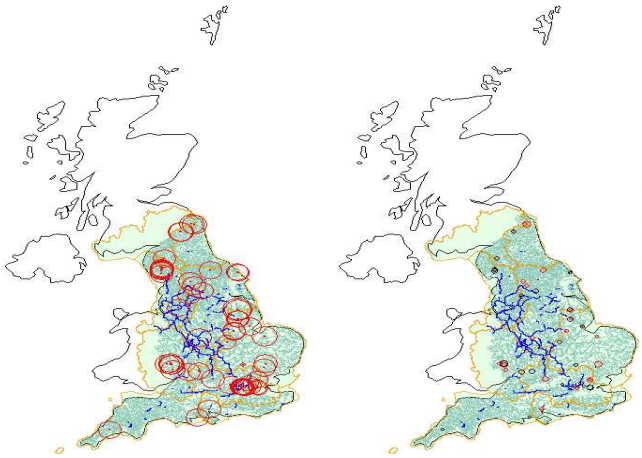

Fig. 9: **Left:** Denoised DO data using 'LG-Did-c-nlt' with 100 trajectories; **Right:** Residual visualisation– positive residuals are represented by red circles, while negative residuals are represented by black circles. The circle size is determined by the residual absolute value.

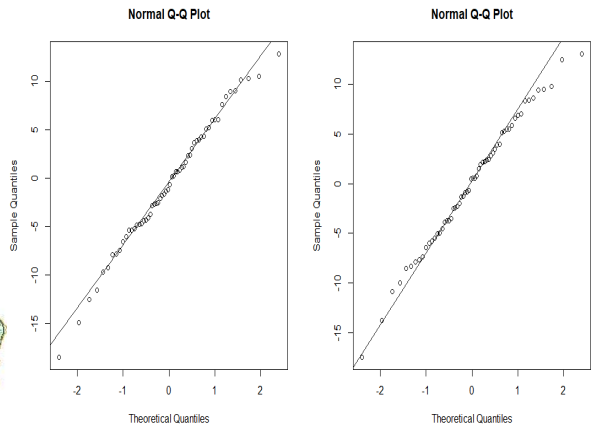

Fig. 10: Residual Q-Q plots for DO data collected on 10/May/2024 (observations from all 60 stations). **Left:** data analysed with proposed 'LG-Aid-c-nlt'. **Right:** analysis using 'LG-Did-c-nlt'.

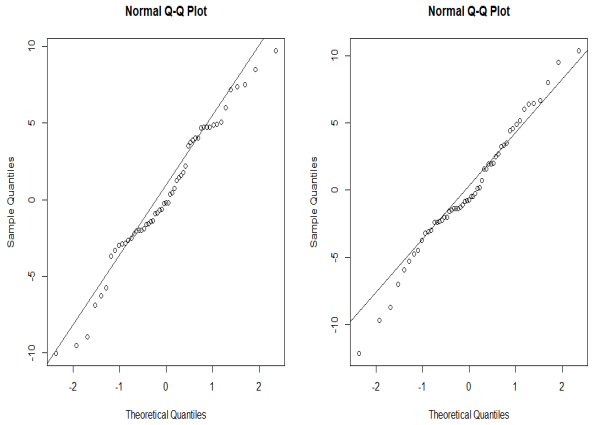

Fig. 11: Residual Q-Q plots for DO data collected on 05/Jun/2024 (55 observations from 60 stations with 5 missing observations). **Left:** 'LG-Aid-c-nlt'. **Right:** 'LG-Did-c-nlt'.

estimate, as the existence of the five missing values on that date resulted in having to artificially connect stations that were not geographically close in order to ensure a connected network and additionally these features were not accounted for since the algorithm used a constant integral initialisation.

On balance, these results give us the confidence that our algorithms work well for this dataset, and more generally the nondecimated 'LG-Aid' appears to be the best choice for hydrological data.

6 Conclusions and further work

Work on hydrological datasets (Park and Oh, 2022) viewed as the analysis of information collected over a network (modelled as a graph), where the observations often pertain to the graph edges (e.g., rivers) rather than to its vertices (e.g., confluences), calls for specifically designed methodology in order to avoid the pitfalls of using noisy edge information as weights, for example. This paper has introduced a novel wavelet-like transformation, LG-LOCAAT, designed to work on network edge-collected data, and has demonstrated its usefulness in the context of an application on denoising a water quality index. Future work could focus on addressing (i) limitations, such as the fact that we rely on a line graph transform with

its associated increased complexity and potential inability to represent the data in its original domain at each stage of the algorithm, (ii) extensions such as, the design of higher order prediction steps that ensure adaptiveness to local smoothness conditions; the development of a framework to allow the computation of confidence intervals; new models that incorporate time-dependency into the noise structure paralleling the GNAR-edge model of Mantziou et al. (2023).

Acknowledgments. Knight and Nason gratefully acknowledge support from EPSRC NeST Programme Grant EP/X002195/1. Cao and Knight gratefully acknowledge that the Viking cluster was used during this project, a high performance computing facility provided by the University of York, UK.

References

Anderes, E., Møller, J., Rasmussen, J.G.: Isotropic covariance functions on graphs and their edges. The Annals of Statistics **48**(4) (2020)

Buisson, L., Blanc, L., Grenouillet, G.: Modelling stream fish species distribution in a river network: the relative effects of temperature versus physical factors. Ecology of Freshwater Fish **17**(2), 244–257 (2008)

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

Bolker, E., Guillemin, V., Holm, T.: How is a graph like a manifold? arXiv preprint math/0206103 (2002)

Bondy, J.A., Murty, U.S.R., *et al.*: Graph Theory with Applications vol. 290. Macmillan London, London (1976)

Bollobás, B.: Modern Graph Theory vol. 184. Springer, New York (2013)

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

Bolin, D., Simas, A., Wallin, J.: Statistical inference for Gaussian Whittle-Matérn fields on metric graphs. arXiv preprint arXiv:2304.10372 (2023)

Bolin, D., Simas, A.B., Wallin, J.: MetricGraph: Random Fields on Metric Graphs. (2023). R package version 1.4.1. https://CRAN.R-project.org/package=MetricGraph

Bolin, D., Simas, A.B., Wallin, J.: Gaussian Whittle-Matérn fields on metric graphs. Bernoulli **30**(2), 1611–1639 (2024)

Chaudhuri, S., Barceló, M.A., Juan, P., Varga, D., Bolin, D., Rue, H., Saez, M.: Enhanced spatial modeling on linear networks using Gaussian Whittle-Matérn fields. Stochastic Environmental Research and Risk Assessment **39**(3), 1143–1158 (2025)

Cohen, A., Daubechies, I., Feauveau, J.-C.: Biorthogonal bases of compactly supported wavelets. Communications on Pure and Applied Mathematics **45**(5), 485–560 (1992)

Cressie, N., Frey, J., Harch, B., Smith, M.: Spatial prediction on a river network. Journal of Agricultural, Biological, and Environmental Statistics **11**, 127–150 (2006)

Chang, J., Kolaczyk, E.D., Yao, Q.: Estimation of subgraph densities in noisy networks. Journal of the American Statistical Association **117**(537), 361–374 (2022)

Chudnovsky, M., Seymour, P.D.: The structure of

claw-free graphs. In: BCC, pp. 153–171 (2005)

Daubechies, I.: Ten Lectures on Wavelets. SIAM, Philadelphia (1992)

Danon, L., Ford, A.P., House, T., Jewell, C.P., Keeling, M.J., Roberts, G.O., Ross, J.V., Vernon, M.C.: Networks and the epidemiology of infectious disease. Interdisciplinary Perspectives on Infectious Diseases **2011**(1), 284909 (2011)

Diestel, R.: Graph theory (3rd ed). Graduate Texts in Mathematics **173**(33), 12 (2005)

Donoho, D.L., Johnstone, J.M.: Ideal spatial adaptation by wavelet shrinkage. Biometrika **81**(3), 425–455 (1994)

Donoho, D.L.: Cart and best-ortho-basis: a connection. The Annals of Statistics **25**(5), 1870–1911 (1997)

Gallacher, K., Miller, C., Scott, E., Willows, R., Pope, L., Douglass, J.: Flow-directed PCA for monitoring networks. Environmetrics **28**(2), 2434 (2017)

Gavish, M., Nadler, B., Coifman, R.R.: Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning. In: ICML (2010)

Higham, N.J.: Accuracy and Stability of Numerical Algorithms. SIAM, Philadelphia (2002)

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

Jansen, M., Bultheel, A.: Smoothing non-equidistantly sampled data using wavelets and cross validation. In: Proceedings of the IEEE Benelux Signal Processing Symposium, pp. 111–114 (1998). Citeseer

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

Jansen, M.H., Oonincx, P.J.: Second Generation Wavelets and Applications. Springer, London (2005)

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

Jackson, M.O., Watts, A.: The evolution of social and economic networks. Journal of Economic Theory **106**(2), 265–295 (2002)

Knight, M., Leeming, K., Nason, G., Nunes, M.: Generalised network autoregressive processes and the GNAR package. Journal of Statistical Software **96**(5), 1–36 (2020)

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

Knight, M.I., Nunes, M.A.: Long memory estimation for complex-valued time series. Statistics and Computing **29**(1), 517–536 (2019)

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

Kuchment, P.: Quantum graphs: I. some basic structures. Waves in Random Media **14**(1), 107 (2003)

Lucasius, C.B., Kateman, G.: Understanding

and using genetic algorithms Part 1. Concepts, properties and context. Chemometrics and Intelligent Laboratory Systems **19**(1), 1–33 (1993)

Lucasius, C.B., Kateman, G.: Understanding and using genetic algorithms Part 2. Representation, configuration and hybridization. Chemometrics and Intelligent Laboratory Systems **25**(2), 99–145 (1994)

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

Mahadevan, N.: Multiscale, multi-dimensional space and space-time function estimation for irregular network data. PhD thesis, University of Bristol (2010)

Mantziou, A., Cucuringu, M., Meirinhos, V., Reinert, G.: The GNAR-edge model: a network autoregressive model for networks with time-varying edge weights. Journal of Complex Networks **11**(6), 039 (2023)

McMillan, H.K., Westerberg, I.K., Krueger, T.: Hydrological data uncertainty and its implications. Wiley Interdisciplinary Reviews: Water **5**(6), 1319 (2018)

Nason, G.P.: Wavelet Methods in Statistics with R vol. 574. Springer, New York (2008)

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

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

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

Roman, S., Axler, S., Gehring, F.: Advanced Linear Algebra vol. 3. Springer, New York (2005)

Severn, K.E., Dryden, I.L., Preston, S.P.: Non-parametric regression for networks. Stat **10**(1), 373 (2021)

Singer, A.: From graph to manifold Laplacian: The convergence rate. Applied and Computational Harmonic Analysis **21**(1), 128–134 (2006)

Stanković, L., Mandic, D., Daković, M., Scalzo, B., Brajović, M., Sejdić, E., Constantinides, A.G.: Vertex-frequency graph signal processing: A comprehensive review. Digital Signal Processing **107**, 102802 (2020)

Shuman, D.I., Ricaud, B., Vandergheynst, P.: Vertex-frequency analysis on graphs. Applied and Computational Harmonic Analysis **40**(2), 260–291 (2016)

Simoens, J., Vandewalle, S.: A stabilized lifting construction of wavelets on irregular meshes on the interval. SIAM Journal on Scientific Computing **24**(4), 1356–1378 (2003)

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

Sweldens, W.: The lifting scheme: A construction of second generation wavelets. SIAM journal on

Mathematical Analysis **29**(2), 511–546 (1998)

Tu, L.W.: Manifolds. In: An Introduction to Manifolds, pp. 47–83. Springer, New York (2011)

Vidakovic, B.: Statistical Modeling by Wavelets vol. 503. John Wiley & Sons, Canada (2009)

Zhou, D., Burges, C.J.: High-order regularization on graphs. In: Proceedings of the 6th International Workshop on Mining and Learning with Graphs (2008)

Zhang, M., Huang, T., Guo, Z., He, Z.: Complex-network-based traffic network analysis and dynamics: A comprehensive review. Physica A: Statistical Mechanics and its Applications **607**, 128063 (2022)

Appendix A Theoretical considerations

A.1 The rationale behind using Euclidean coordinates

Many works in the literature explore the connection between graphs and manifolds (Bolker et al., 2002; Singer, 2006). Zhou and Burges (2008) show that a graph can be considered as the discrete approximation of a manifold, or a manifold can be treated as the continuous analogue of a graph.

A manifold is a topological space that is locally homeomorphic to an Euclidean space $\mathbb{R}^p$ for some p. Hence, any point on the manifold has a neighbourhood that can be mapped onto an open set of $\mathbb{R}^p$ (Tu, 2011). Therefore, although a manifold is a space naturally without coordinates, one may often use coordinate information for its local structures. Consider the toy graph in Figure A1, which contains six vertices $\{v_i\}_{i=1}^6$. The points $\{\mathbf{p}_k\}_{k=1}^5$ on the edges represent the metrized locations of the observations and we denote by e_k the edge associated to $\mathbf{p}_k$ and by U_k the open interval corresponding to the k-th edge (see Section 2.1). The open set associated to incident edges (for example, edges 1, 2 and 3) can be considered as a homeomorphism of a continuous open set in a Euclidean space. Hence, the Euclidean distance (for example, among $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$) supplies local structure information when performing the line graph transformation.

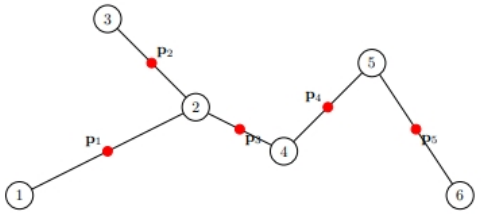

Fig. A1: A toy tree graph with six vertices.

A.2 Sparsity

In the context of wavelet transforms, the term 'sparsity' usually indicates that the wavelet (detail) coefficients form a sparse sequence which represents the original signal energy by concentrating it in a small amount of coefficients. The properties of the signal are of course an essential factor in achieving sparsity, as well as the traits of the wavelet transform. In our context, recall our focus is on representing in the lifting domain a function g^{Γ^*} defined on the metric space (Γ^*, dist), where the metrized locations $\{\mathbf{p}^*_{v^*_k}\}_{k=1}^m \in \Gamma^*$ are associated with the line graph vertices $\{v^*_k\}_{k=1}^m \in \mathcal{V}^*$. Then a Euclidean analogue of Lipschitz continuity on a graph can be generalised as follows.

Definition 1. *A function g^{Γ^*} defined on Γ^* has a point $\mathbf{p}^*_L$ of Lipschitz continuity if there exists an interval $\mathbf{I}^* \subset \Gamma^*$, such that for all $\mathbf{p}^* \in \mathbf{I}^*$,*

$$\left|g^{\Gamma^*}(\mathbf{p}^*_L) - g^{\Gamma^*}(\mathbf{p}^*)\right| \leq C\, dist(\mathbf{p}^*_L, \mathbf{p}^*),$$

where $0 < C < \infty$ is a constant.

The spatial Lipschitz continuity used in Jansen et al. (2009) and the Hölder class for tree graphs defined by Gavish et al. (2010) motivate this definition, which allows a bound to be established on the detail coefficients, as described next.

Proposition 2. *For a stage-r prediction step, if for all $v_k^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r} \cup \{v^*_{k_r}\}$, we have $c^{\Gamma^*}_{k,r} = g^{\Gamma^*}(\mathbf{p}^*_{v^*_k})$, and g^{Γ^*} is Lipschitz continuous over an interval that contains the metrized points associated with $\mathcal{N}^{\mathcal{V}^*}_{k_r,r}$, then the detail coefficient obtained by equation (16) at stage-r satisfies that*

$$\left|d^{\Gamma^*}_{k_r}\right| \leq C \, \frac{\sum_{s:v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} dist(\mathbf{p}^*_{v^*_{k_r}}, \mathbf{p}^*_{v^*_s})}{|\mathcal{N}^{\mathcal{V}^*}_{k_r,r}|}. \tag{A1}$$

For the the proof, the reader can refer to Appendix B. Nonetheless, acquiring a precise bound for all detail coefficients is not a straightforward task for iterative methods. Let us consider the following situation, suppose at stage-r, the function underpinning the scaling coefficients $(c^{\Gamma*}_{k,r})$ is Lipschitz continuous, which indicates that

$$\left|c^{\Gamma^*}_{k,r} - c^{\Gamma^*}_{s,r}\right| \leq C \, \mathrm{dist}(\mathbf{p}^*_{v^*_{k_r}}, \mathbf{p}^*_{v^*_s}).$$

Assuming $v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$, and $v^*_k \notin \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$, then from stage-$r$ to stage-$(r-1)$, we have

$$\begin{aligned} c^{\Gamma^*}_{k,r-1} &= c^{\Gamma^*}_{k,r}, \\ c^{\Gamma^*}_{s,r-1} &= c^{\Gamma^*}_{s,r} + b^{\Gamma^*}_{s,r} d^{\Gamma^*}_{k_r}. \end{aligned}$$

Under the assumptions of Proposition 2, the bound for the absolute difference between $c^{\Gamma^*}_{k,r-1}$ and $c^{\Gamma^*}_{s,r-1}$ then becomes

$$\begin{aligned} \left|c^{\Gamma^*}_{k,r-1} - c^{\Gamma^*}_{s,r-1}\right| &= \left|\left(c^{\Gamma^*}_{k,r} - c^{\Gamma^*}_{s,r}\right) - b^{\Gamma^*}_{s,r} d^{\Gamma^*}_{k_r}\right| \\ &\leq \left|c^{\Gamma^*}_{k,r} - c^{\Gamma^*}_{s,r}\right| + b^{\Gamma^*}_{s,r}\left|d^{\Gamma^*}_{k_r}\right| \\ &\leq C \, \mathrm{dist}(\mathbf{p}^*_{v^*_{k_r}}, \mathbf{p}^*_{v^*_s}) \\ &\quad + b^{\Gamma^*}_{s,r} C \, \frac{\sum_{t:v^*_t \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} \mathrm{dist}(\mathbf{p}^*_{v^*_{k_r}}, \mathbf{p}^*_{v^*_t})}{|\mathcal{N}^{\mathcal{V}^*}_{k_r,r}|}. \end{aligned}$$

Hence, there is no guarantee that the function underpinning $c^{\Gamma^*}_{k,r-1}$ is still Lipschitz continuous with the same constant C. This indicates that even if $v^*_{k_r}$ and $v^*_{k_{r-1}}$ are 'close' to each other for some stage-r, unwanted effects on the compression ability of the algorithm may still occur.

A.3 Stability

As wavelet functions generated by the lifting scheme (including LOCAAT-based approaches) are no longer orthogonal, the algorithm stability may become an issue. One way to guarantee the stability of the transform is to ensure that both primal and dual wavelet functions form Riesz bases, such that

$$L \, \|g^{\Gamma^*}\|^2_{L_2} \leq \sum_{k \in \mathcal{D}_r} |\langle g^{\Gamma^*}, \psi^{\Gamma^*}_k \rangle|^2 \leq U \, \|g^{\Gamma^*}\|^2_{L_2}, \tag{A2}$$

$$\tilde{L} \, \|g^{\Gamma^*}\|^2_{L_2} \leq \sum_{k \in \mathcal{D}_r} |\langle g^{\Gamma^*}, \tilde{\psi}^{\Gamma^*}_k \rangle|^2 \leq \tilde{U} \, \|g^{\Gamma^*}\|^2_{L_2}, \tag{A3}$$

where $\mathcal{D}_r = \{k_m, ..., k_{r+1}\}$. If we can find the upper bounds, then the lower bound can be obtained by duality, such that $L = \tilde{U}^{-1}$ and $\tilde{L} = U^{-1}$ holds, see Cohen et al. (1992) and Daubechies (1992). However, as pointed out by Jansen et al. (2009), it is challenging to verify whether a set of bases satisfies the Riesz condition on a global scale, especially in irregular scenarios. Simoens and Vandewalle (2003) and Jansen and Oonincx (2005) presented a necessary but not sufficient condition for the Riesz condition is that each one-step transform and its inverse are uniformly bounded. This can be articulated as the one-level lifting operator and its inverse being bounded, see Simoens and Vandewalle (2003). For the LOCAAT-based algorithm, it means that the quantities resulting from the predict (equation (16)), the update (equation (19)), undo update

(equation (21)), and undo predict (equation (22)) should be bounded in norm. For the forward prediction, given that $\sum_{s:v_k^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} = 1$ and $a^{\Gamma^*}_{s,r} \geq 0$, we have that

$$\begin{aligned} |d^{\Gamma^*}_{k_r}|^2 &= |c^{\Gamma^*}_{k_r,r} - \sum_{s:v_s^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} c^{\Gamma^*}_{s,r}|^2 \\ &\leq (1 + \sum_{s:v_s^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} |a^{\Gamma^*}_{s,r}|^2) \sum_{k:v_k^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r} \cup \{k_r\}} |c^{\Gamma^*}_{k,r}|^2, \\ &\leq 2 \sum_{k:v_k^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r} \cup \{k_r\}} |c^{\Gamma^*}_{k,r}|^2. \end{aligned} \tag{A4}$$

by the Cauchy-Schwarz inequality. With update filters satisfying $0 < b^{\Gamma^*}_{s,r} \leq \frac{1}{2}$, for all $v_s^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$ (Jansen et al., 2009), this guarantees that $c^{\Gamma^*}_{s,r-1}$ is bounded after the update (equation (19)) for all $v_s^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$. For the one-level undo lifting, the boundness can be obtained by duality, see Jansen et al. (2009). From equation (A4), note that the upper bound is given by the value $(1 + \sum_{s:v_s^* \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} |a^{\Gamma^*}_{s,r}|^2)$. Hence, if we lift a new vertex which only has one neighbouring new vertex, the bound tends to be the maximum (2). This observation matches the practical sensitive points phenomena observed by Jansen and Bultheel (1998) and Mahadevan (2010), where the term 'sensitive points' indicates that boundary points contain high energy and have a significant influence on recovering the signal. On the other hand, if a new vertex with a large neighbourhood has been lifted and the prediction weights are almost evenly distributed (e.g. moving average), then the upper bound tends to be small.

A.4 Scale Interpretation

The notion of scale is essential for any wavelet-based method and relates to the level of detail, also known as the resolution, while also being strongly connected with the Fourier-based frequency (Nason, 2008; Vidakovic, 2009). While in the classical wavelet methods the scales can be generated by the dilation relation, within the LOCAAT framework the concept of scale is not directly discernible due to the absence of such relation. Inspired by the classical scale concept, the quantities in equation (A1) can help in establishing a 'scale'. Note that its expression is very similar to the 'sum of distances' and the 'average distance integral', thus, we simply define the scale of the detail coefficient $d^{\Gamma^*}_{k_r}$ obtained at stage-r as

$$\text{scale}^{\Gamma^*}_{k_r} = I^{\Gamma^*}_{k_r,r} \tag{A5}$$

The integral satisfies that $I^{\Gamma^*}_{k_{r-1},r-1} \geq I^{\Gamma^*}_{k_r,r}$, which implies the correspondence between removal order and scale. Note that when using the integral values as a sequence of ones, the potential advantage of this integral choice is that for the initial lifting recursions, the next stage-$(r-1)$ removal choice $v^*_{k_{r-1}}$ cannot be a neighbouring vertex of $v^*_{k_r}$. For example, once we remove $v^*_{k_m}$, after the integral update, the integral values of its neighbourhood will exceed the value one, which guarantees that another vertex (not in the neighbourhood) will be picked for removal, hence the network space is explored more efficiently.

A.5 Original Domain Transformation

While the overall LG-LOCAAT transform and the line graph mapping (from G to G^*) are invertible, the existence of a line graph inverse cannot be guaranteed at every stage r. This is since while any graph has its unique corresponding line graph, not all graphs are line graphs (Bondy et al., 1976).

As an example, let us consider the case where we lifted a new vertex $v^*_{k_r}$ at stage-r, and its neighbourhood is denoted as $\mathcal{N}^{\mathcal{V}^*}_{k_r,r}$. Therefore, we have the subgraph $G^{*\text{supp}}_r = (\mathcal{V}^{*\text{supp}}_r, \mathcal{E}^{*\text{supp}}_r)$, where $\mathcal{V}^{*\text{supp}}_r = \{v^*_{k_r}\} \cup \mathcal{N}^{\mathcal{V}^*}_{k_r,r}$ and $\mathcal{E}^{*\text{supp}}_r =$

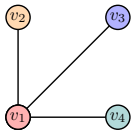

Fig. A2: An example of a claw graph.

$\{\{v^*_{k_r}, v^*_s\}\}_{s:\, v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}}$. Recall that the prediction step at stage-r is performed on this subgraph $G^{*\text{supp}}_r$, hence the analytic form of the wavelet function $\varphi^{\Gamma^*}_{k_r}$ is defined on the topology of $G^{*\text{supp}}_r$. Now if we want to obtain the associated analytical form defined on the original graph G for $\varphi^{\Gamma^*}_{k_r}$, we have to find the inverse of $G^{*\text{supp}}_r$. Let us consider a special case, suppose there are three components in the neighbourhood $\mathcal{N}^{\mathcal{V}^*}_{k_r,r}$. This graph topology is referred to as 'claw graph' in graph theory literature, see Bondy et al. (1976) and Figure A2. Unfortunately, claw graphs are not line graphs (Chudnovsky and Seymour, 2005), which indicates that $G^{*\text{supp}}_r$ has no interpretation in the original graph domain if it is a claw graph. Moreover, any star graph with more than four vertices is not a line graph (Bondy et al., 1976). Hence, at any stage-r, if there are more than three vertices used for prediction, then the associated topology $G^{*\text{supp}}_r$ will have no interpretation. This indicates that we cannot guarantee a graph correspondence between the original and line graph domains at each stage of the algorithm.

Appendix B Proofs

Proof for Proposition 1

Starting with $C \cdot \underline{I}^* = \{CI^*_{k,m}\}_{k\in\{1,\dots m\}}$ will not change the stage-m split step as long as $C > 0$, and the same new vertex $v^*_{k_m}$ will be lifted at this stage. Since the predict step is independent of the integral values, the value of the detail coefficient $d^*_{k_m}$ and the prediction weights remain the same. For any $s \in \mathcal{N}^{\mathcal{V}^*}_{k_m,m}$, the update of the integral sequence is now

$$\begin{aligned} &CI^*_{s,m} + a^*_{s,m} CI^*_{k_m,m} \\ =&C(I^*_{s,m} + a^*_{s,m} I^*_{k_m,m}) \\ =&CI^*_{s,m-1}, \end{aligned}$$

which indicates that in the stage-$(m-1)$, the integral sequence is proportional to the integral with the same multiplier constant C. The s-th update coefficient is determined by the minimum norm solution such that

$$\begin{aligned} b^{\Gamma^*}_{s,m} &= \frac{CI^*_{s,m-1} CI^*_{k_m,m}}{\sum_{t:v^*_t \in \mathcal{N}^{\mathcal{V}^*}_{k_m,m}} (CI^*_{t,m-1})^2} \\ &= \frac{I^*_{s,m-1} I^*_{k_m,m}}{\sum_{t:v^*_t \in \mathcal{N}^{\mathcal{V}^*}_{k_m,m}} (I^*_{t,m-1})^2}, \end{aligned}$$

which coincides with the result when starting with $\underline{I}^*$. Thus, repeating the procedures above and by induction, we conclude that the detail coefficients and the filters do not change upon a proportional change in the integral values.

Proof for Proposition 2

Recall the detail coefficient is obtained by the prediction step, in which we have

$$\left| d^{\Gamma^*}_{k_r} \right| = \left| c^{\Gamma^*}_{k_r,r} - \sum_{s:v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} c^{\Gamma^*}_{s,r} \right|. \tag{B6}$$

If for all $t : v^*_t \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r} \cup \{v^*_{k_r}\}$, we have that $c^{\Gamma^*}_{t,r} = g^{\Gamma^*}_t$, then equation (B6) can be written as

$$\begin{aligned} \left| d^{\Gamma^*}_{k_r} \right| &= \left| g^{\Gamma^*}_{k_r} - \sum_{s:v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} g^{\Gamma^*}_s \right| \\ &= \left| \sum_{s:v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} (g^{\Gamma^*}_{k_r} - g^{\Gamma^*}_s) \right| \\ &\le \sum_{s:v^*_s \in \mathcal{N}^{\mathcal{V}^*}_{k_r,r}} a^{\Gamma^*}_{s,r} \left| g^{\Gamma^*}_{k_r} - g^{\Gamma^*}_s \right|, \end{aligned} \tag{B7}$$

since $a_{s,r}^{\Gamma^*} \geq 0$ for all $s : v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}$. Then because the function g^{Γ^*} is Lipschitz with a constant $0 < C < \infty$, then we have

$$\begin{aligned}\left|d_{k_r}^{\Gamma^*}\right| &\leq \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} a_{s,r}^{\Gamma^*} \left|g_{k_r}^{\Gamma^*} - g_s^{\Gamma^*}\right| \\ &\leq \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} a_{s,r}^{\Gamma^*} C \operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_s^*}^*). \end{aligned} \tag{B8}$$

Recall that the prediction weights are normalised inverse distances, $a_{s,r}^{\Gamma^*} = \frac{1/\operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_s^*}^*)}{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} 1/\operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)}$. Plug it into the inequality (B8), we have

$$\begin{aligned}\left|d_{k_r}^{\Gamma^*}\right| &\leq \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} \frac{1/\operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_s^*}^*)}{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} 1/\operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)} C \operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_s^*}^*) \\ &= \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} \frac{1}{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} 1/\operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)} C \\ &= \frac{1}{|\mathcal{N}_{k_r,r}^{\mathcal{V}^*}|} \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} \frac{|\mathcal{N}_{k_r,r}^{\mathcal{V}^*}|}{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} 1/\operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)} C \\ &\leq \frac{1}{|\mathcal{N}_{k_r,r}^{\mathcal{V}^*}|} \sum_{s:v_s^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} \frac{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} \operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)}{|\mathcal{N}_{k_r,r}^{\mathcal{V}^*}|} C \\ &= \frac{\sum_{t:v_t^* \in \mathcal{N}_{k_r,r}^{\mathcal{V}^*}} \operatorname{dist}(\mathbf{p}_{v_{k_r}^*}^*, \mathbf{p}_{v_t^*}^*)}{|\mathcal{N}_{k_r,r}^{\mathcal{V}^*}|} C, \end{aligned} \tag{B9}$$

since for positive $x_1, ..., x_n$, we have

$$\frac{n}{1/x_1 + ... + 1/x_n} \leq \frac{x_1 + ... + x_n}{n}. \tag{B10}$$

Appendix C Supporting evidence for Section 4.3

C.1 Stability Results

Tables C1 and C2 show the quantile of the condition number of the lifting matrices constructed via the simulation study. All algorithms exhibit similar condition numbers, with a preference for initial integrals given as a sequence of ones and for simple moving average prediction weights.

C.2 Sparsity Plots

Figures C3 and C4 illustrate the sparsity plots of the test functions in Figure 3, and follow the pointwise construction as described for Figure 4, while Figures C5 and C6 show those functions that follow the edge averaging construction as described for Figure 5.

C.3 Denoising Performance Tables

Tables C3–C6 show the variance and the squared bias for our simulation study. The tables show that the use of average distance as integrals typically leads to lower variances, while the use of a sequence of ones as integrals typically leads to lower squared bias.

Appendix D Supporting evidence for Section 5.1

Condition Number	Max	75%	Median	25%	Min
LG-Sid-c	14.5314	12.9183	12.4962	11.9702	11.1918
LG-Aid-c	15.0632	13.2504	12.5252	11.9598	11.1996
LG-Did-c	13.9051	12.5774	11.5010	11.0700	10.6420
LG-Snw-c	13.5717	12.3798	11.7343	11.3743	10.8643
LG-Anw-c	12.7559	11.4412	11.0405	10.5475	10.0281
LG-Dnw-c	**12.1607**	**11.0283**	**10.5684**	**10.2285**	**9.9500**

Table C1: Condition number for LG-LOCAAT with coordinate information.

Condition Number	Max	75%	Median	25%	Min
LG-Sid-p	13.4308	12.3485	11.7877	11.3759	10.7340
LG-Aid-p	12.7611	11.4225	10.9914	10.5397	10.0863
LG-Did-p	12.6918	11.6651	10.7789	10.3077	10.0251
LG-Snw-p	13.2506	12.0824	11.5843	11.2258	10.6871
LG-Anw-p	12.5608	11.3372	10.7768	10.3567	10.0530
LG-Dnw-p	**12.2235**	**11.0813**	**10.5528**	**10.1628**	**9.9535**

Table C2: Condition number for LG-LOCAAT using the path length.

Variance$\times 10^3$	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-c	48	65	61	51	68	39
LG-Aid-c	**46**	**64**	**55**	**48**	**64**	**36**
LG-Did-c	49	81	66	56	101	37
LG-Snw-c	49	66	61	52	69	39
LG-Anw-c	**46**	**64**	**55**	**48**	**64**	**36**
LG-Dnw-c	52	86	71	60	113	39
SNR=5						
LG-Sid-c	**18**	**24**	23	22	31	15
LG-Aid-c	**18**	**24**	22	**20**	27	**14**
LG-Did-c	20	33	27	26	67	**15**
LG-Snw-c	19	25	23	22	31	15
LG-Anw-c	**18**	**24**	**21**	**20**	**26**	**14**
LG-Dnw-c	21	34	30	28	78	17
SNR=7						
LG-Sid-c	**9**	**13**	**12**	12	18	9
LG-Aid-c	**9**	**13**	**12**	**11**	**15**	**8**
LG-Did-c	10	18	15	16	57	9
LG-Snw-c	**9**	**13**	**12**	12	18	9
LG-Anw-c	**9**	**13**	**12**	12	**15**	**8**
LG-Dnw-c	10	18	16	17	68	10

Table C3: Variance for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the pointwise construction. We assume the coordinate information is available.

Bias$^2 \times 10^3$	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-c	17	49	48	41	204	13
LG-Aid-c	17	49	37	**32**	**134**	9
LG-Did-c	**14**	**39**	**27**	24	**134**	**8**
LG-Snw-c	17	51	48	44	213	14
LG-Anw-c	18	51	37	33	141	10
LG-Dnw-c	15	40	**27**	**23**	148	**8**
SNR=5						
LG-Sid-c	4	17	20	23	175	10
LG-Aid-c	5	18	16	17	112	7
LG-Did-c	**3**	**13**	**10**	**13**	**111**	**6**
LG-Snw-c	4	17	21	25	186	11
LG-Anw-c	5	19	17	18	121	8
LG-Dnw-c	**3**	**13**	**10**	**13**	129	**6**
SNR=7						
LG-Sid-c	2	9	12	16	166	9
LG-Aid-c	2	9	9	11	105	**6**
LG-Did-c	**1**	**7**	**5**	**8**	**104**	**5**
LG-Snw-c	**1**	9	12	16	177	9
LG-Anw-c	2	10	10	12	115	6
LG-Dnw-c	**1**	**7**	**5**	**8**	124	**5**

Table C4: Squared bias for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the pointwise construction. We assume the coordinate information is available.

Variance$\times 10^3$	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-p	49	66	61	52	69	39
LG-Aid-p	**47**	**64**	57	**49**	**63**	**36**
LG-Did-p	51	83	69	60	113	39
LG-Snw-p	49	67	61	52	70	39
LG-Anw-p	**47**	65	**56**	50	64	37
LG-Dnw-p	53	88	73	63	121	40
SNR=5						
LG-Sid-p	19	25	23	22	31	16
LG-Aid-p	**18**	**24**	**22**	**21**	**26**	**15**
LG-Did-p	21	34	29	28	77	17
LG-Snw-p	19	25	23	22	31	16
LG-Anw-p	**18**	25	**22**	**21**	**26**	**15**
LG-Dnw-p	21	35	30	29	85	18
SNR=7						
LG-Sid-p	**9**	**13**	12	**12**	18	9
LG-Aid-p	**9**	**13**	**11**	**12**	**15**	**8**
LG-Did-p	10	18	16	17	65	10
LG-Snw-p	**9**	**13**	12	13	19	9
LG-Anw-p	**9**	**13**	12	**12**	**15**	**8**
LG-Dnw-p	10	19	17	18	74	11

Table C5: Variance for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the pointwise construction. The path distance is used.

Bias^2	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-p	17	50	48	42	205	14
LG-Aid-p	18	53	37	35	145	11
LG-Did-p	**14**	**39**	**26**	**23**	**140**	**8**
LG-Snw-p	19	51	49	43	218	15
LG-Anw-p	19	58	39	35	151	11
LG-Dnw-p	15	41	27	**23**	155	**8**
SNR=5						
LG-Sid-p	4	17	21	24	177	11
LG-Aid-p	5	20	16	19	126	8
LG-Did-p	**3**	**13**	**9**	**12**	**115**	**6**
LG-Snw-p	4	17	21	25	188	12
LG-Anw-p	5	21	17	19	132	9
LG-Dnw-p	**3**	**13**	**9**	**12**	133	**6**
SNR=7						
LG-Sid-p	2	9	12	16	168	9
LG-Aid-p	2	10	10	12	120	6
LG-Did-p	**1**	**7**	**5**	**8**	**107**	**5**
LG-Snw-p	2	9	12	16	178	9
LG-Anw-p	2	11	10	13	127	7
LG-Dnw-p	**1**	**7**	**5**	**8**	127	**5**

Table C6: Squared bias for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the pointwise construction. The path distance is used.

Variance$\times 10^3$	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-c	45	65	56	51	67	38
LG-Aid-c	**42**	**62**	52	**47**	**64**	**36**
LG-Did-c	46	78	60	55	99	37
LG-Snw-c	46	65	56	52	67	38
LG-Anw-c	**42**	63	**51**	48	**64**	**36**
LG-Dnw-c	48	84	65	59	111	39
SNR=5						
LG-Sid-c	18	26	23	22	30	15
LG-Aid-c	**17**	**25**	**21**	**20**	27	**14**
LG-Did-c	20	35	27	26	65	15
LG-Snw-c	18	26	23	22	30	15
LG-Anw-c	18	26	**21**	**20**	**26**	**14**
LG-Dnw-c	21	37	29	28	77	17
SNR=7						
LG-Sid-c	10	14	**12**	12	18	8
LG-Aid-c	**9**	**13**	**12**	**11**	**15**	8
LG-Did-c	11	19	15	16	55	9
LG-Snw-c	10	14	**12**	12	17	9
LG-Anw-c	**9**	**13**	**12**	**11**	**15**	8
LG-Dnw-c	11	21	17	17	67	10

Table C7: Variance for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the edge-averaging construction. We assume the coordinate information is available.

Bias^2	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-c	19	61	47	41	215	12
LG-Aid-c	17	58	37	31	142	9
LG-Did-c	**13**	**42**	**28**	**23**	**130**	**7**
LG-Snw-c	20	63	48	44	224	13
LG-Anw-c	18	62	39	32	152	9
LG-Dnw-c	**13**	44	**28**	**23**	144	**7**
SNR=5						
LG-Sid-c	7	25	22	24	190	10
LG-Aid-c	7	24	18	18	120	7
LG-Did-c	**5**	**17**	**11**	**12**	**111**	**6**
LG-Snw-c	7	25	23	25	201	11
LG-Anw-c	8	25	19	19	131	7
LG-Dnw-c	**5**	**17**	**11**	**12**	128	**6**
SNR=7						
LG-Sid-c	3	13	14	16	181	9
LG-Aid-c	3	13	11	12	114	6
LG-Did-c	**2**	**9**	**6**	**8**	**105**	**4**
LG-Snw-c	3	14	14	17	193	9
LG-Anw-c	4	14	11	12	126	6
LG-Dnw-c	**2**	**9**	**6**	**8**	123	5

Table C8: Squared bias for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the edge-averaging construction. We assume the coordinate information is available.

Variance$\times 10^3$	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-p	46	65	56	52	67	39
LG-Aid-p	**43**	**63**	53	**49**	**63**	**36**
LG-Did-p	48	81	63	59	111	39
LG-Snw-p	46	65	56	52	69	39
LG-Anw-p	**43**	64	**52**	**49**	64	**36**
LG-Dnw-p	49	86	67	62	119	40
SNR=5						
LG-Sid-p	19	26	23	22	30	**15**
LG-Aid-p	**18**	**25**	**21**	**21**	**26**	**15**
LG-Did-p	20	36	29	28	76	17
LG-Snw-p	19	26	23	22	30	16
LG-Anw-p	**18**	26	22	**21**	**26**	**15**
LG-Dnw-p	21	38	30	29	84	18
SNR=7						
LG-Sid-p	10	**14**	**12**	**12**	18	9
LG-Aid-p	**9**	**14**	**12**	**12**	**15**	**8**
LG-Did-p	11	20	16	17	65	10
LG-Snw-p	10	**14**	13	**12**	18	9
LG-Anw-p	10	**14**	**12**	**12**	**15**	**8**
LG-Dnw-p	12	21	17	18	74	11

Table C9: Variance for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the edge-averaging construction. The path distance is used.

Bias^2	g_1	Blocks	Doppler	Bumps	Heavisine	mfc
SNR=3						
LG-Sid-p	20	62	47	42	215	14
LG-Aid-p	18	61	38	34	155	10
LG-Did-p	**13**	**41**	**27**	**23**	**136**	**7**
LG-Snw-p	20	64	48	44	227	14
LG-Anw-p	20	69	40	35	162	10
LG-Dnw-p	**13**	44	28	**23**	151	**7**
SNR=5						
LG-Sid-p	8	25	22	24	192	11
LG-Aid-p	8	26	18	19	135	8
LG-Did-p	**4**	**16**	**11**	**12**	**115**	**6**
LG-Snw-p	8	26	23	25	203	11
LG-Anw-p	8	29	19	19	144	8
LG-Dnw-p	5	17	**11**	**12**	133	**6**
SNR=7						
LG-Sid-p	3	14	13	16	185	9
LG-Aid-p	4	15	11	12	130	6
LG-Did-p	**2**	**9**	**6**	**7**	**108**	**4**
LG-Snw-p	3	14	14	17	194	9
LG-Anw-p	4	15	12	13	138	6
LG-Dnw-p	**2**	**9**	**6**	8	127	5

Table C10: Squared bias for LG-LOCAAT on a tree structure with 100 nodes and 99 edges. The functions follow the edge-averaging construction. The path distance is used.

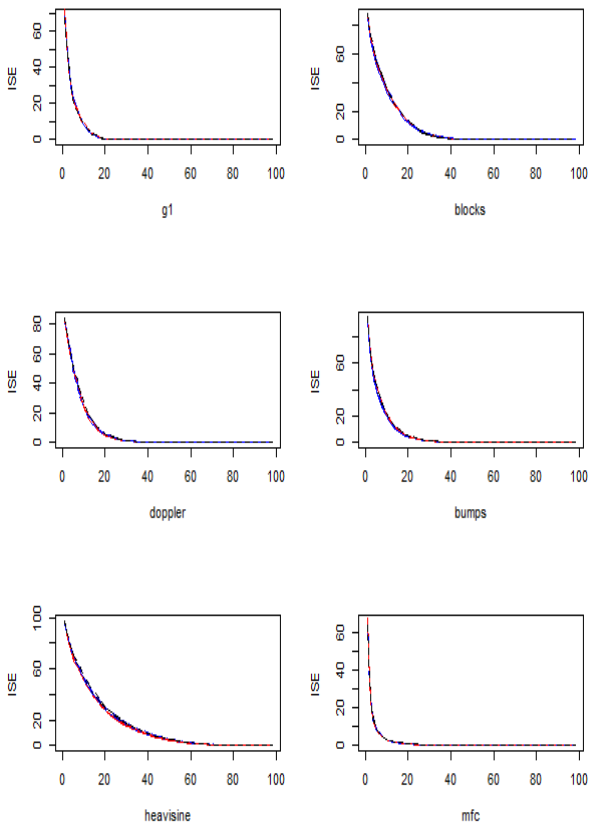

Fig. C3: Sparsity plots for the test functions in equation (25). The scheme is based on coordinate information. From left to right on *top row*: g_1, Blocks; *middle row*: Doppler, Bumps; *bottom row*: Heavisine, maartenfunc. **Black line**: LG-Sid-c; **red line**: LG-Aid-c; **blue line**: LG-Did-c; **dashed black line**: LG-Snw-c; **dashed red line**: LG-Anw-c; **dashed blue line**: LG-Dnw-c.

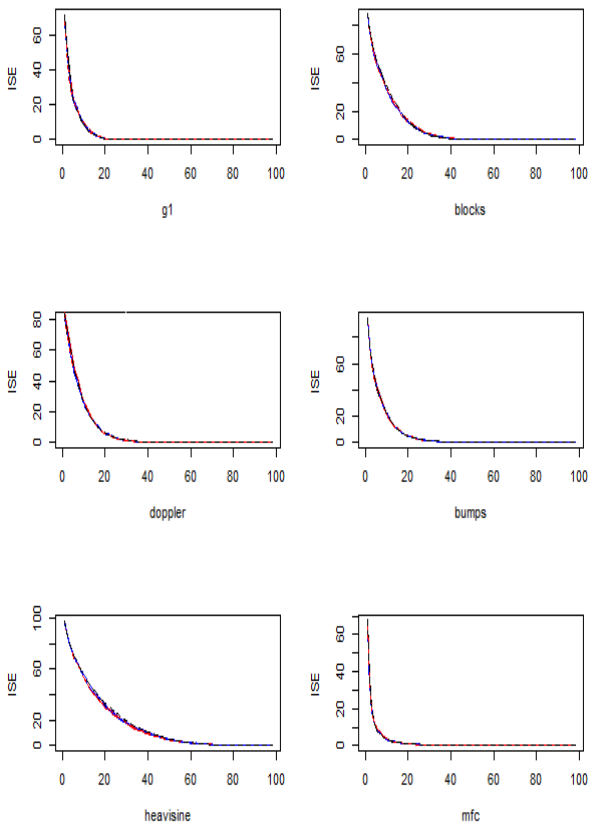

Fig. C4: Sparsity plots for the test functions in equation (25). The scheme is based on path distance. From left to right on *top row*: g_1, Blocks; *middle row*: Doppler, Bumps; *bottom row*: Heavisine, maartenfunc. **Black line**: LG-Sid-p; **red line**: LG-Aid-p; **blue line**: LG-Did-p; **dashed black line**: LG-Snw-p; **dashed red line**: LG-Anw-p; **dashed blue line**: LG-Dnw-p.

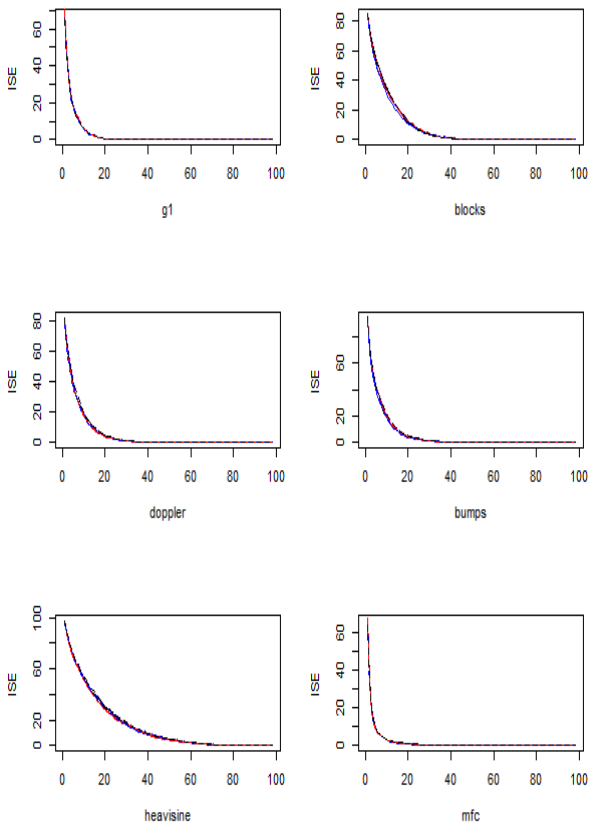

Fig. C5: Sparsity plots for the test functions in equation (26). The scheme is based on coordinate information. From left to right on *top row*: g_1, Blocks; *middle row*: Doppler, Bumps; *bottom row*: Heavisine, maartenfunc. **Black line**: LG-Sid-c; **red line**: LG-Aid-c; **blue line**: LG-Did-c; **dashed black line**: LG-Snw-c; **dashed red line**: LG-Anw-c; **dashed blue line**: LG-Dnw-c.

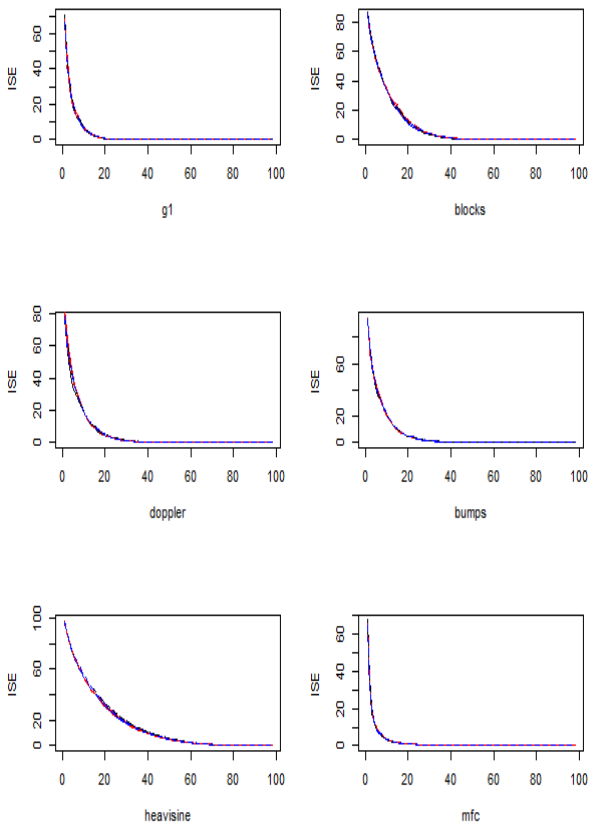

Fig. C6: Sparsity plots for the test functions in equation (26). The scheme is based on path distance. From left to right on *top row*: g_1, Blocks; *middle row*: Doppler, Bumps; *bottom row*: Heavisine, maartenfunc. **Black line**: LG-Sid-p; **red line**: LG-Aid-p; **blue line**: LG-Did-p; **dashed black line**: LG-Snw-p; **dashed red line**: LG-Anw-p; **dashed blue line**: LG-Dnw-p.

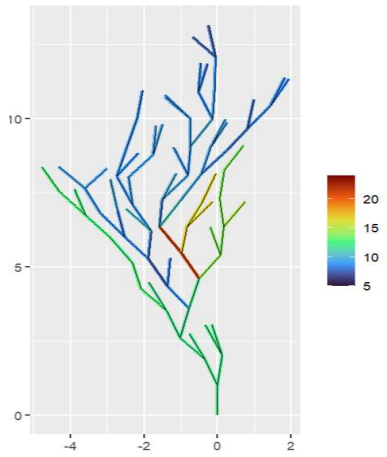

Fig. D7: The denoised river flow data by means of 'LG-Aid-p-nlt' with 30 trajectories.