2019-03-21

# Distributed Adaptive Estimation of Non-Stationary Signal and Outliers in Time-Varying Graphs

May Zar Lin
*University of Miami*, mayzarlin.ec@gmail.com

UNIVERSITY OF MIAMI

# DISTRIBUTED ADAPTIVE ESTIMATION OF NON-STATIONARY SIGNAL AND OUTLIERS IN TIME-VARYING GRAPHS

By

May Zar Lin

A DISSERTATION

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Coral Gables, Florida

May 2019

UNIVERSITY OF MIAMI

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

# DISTRIBUTED ADAPTIVE ESTIMATION OF NON-STATIONARY SIGNAL AND OUTLIERS IN TIME-VARYING GRAPHS

May Zar Lin

Approved:

| | |
|---|---|
| Manohar N. Murthi, Ph.D.<br>Associate Professor of Electrical and Computer Engineering | Kamal Premaratne, Ph.D.<br>Professor of Electrical and Computer Engineering |
| Xiaodong Cai, Ph.D.<br>Professor of Electrical and Computer Engineering | Jie Xu, Ph.D.<br>Assistant Professor of Electrical and Computer Engineering |
| Odelia Schwartz, Ph.D.<br>Associate Professor of Computer Science | Guillermo J. Prado, Ph.D.<br>Dean of the Graduate School |

LIN, MAY ZAR (Ph.D., Electrical and Computer Engineering)

Distributed Adaptive Estimation of Non-Stationary Signal and Outliers in Time-Varying Graphs (May 2019)

Abstract of a dissertation at the University of Miami.

Dissertation supervised by Associate Professor Manohar N. Murthi.
No. of pages in text. (116)

In sensor networks, adaptive algorithms such as diffusion adaptation LMS and RLS are commonly used to learn and track non-stationary signals. When such signals have similarities across certain nodes as captured by a graph, then Laplacian Regularized (LR) LMS and diffusion adaptation LR LMS can be utilized for the respective centralized and distributed estimation cases. What if the ground truth signal's time-varying co-variance structure is related to a time-varying graph? And what if there exists outlier/anomaly nodes trying to influence the graph signal?

In order to answer these questions, we first re-examine the existing adaptive methods, and use graph signal processing notions to augment the algorithms with an additional graph filtering step for regularization. We also study a distributed adaptive algorithm based on message passing that does not require any global information and scales to large time-varying graphs. In particular, each node augments adaptive filtering steps with an additional local filtering steps based on a Local Graph Transform (LGT) defined by the particular node's local graph Laplaican.

Moreover, we demonstrate how to design these graph filters, leading to performance improvements over existing methods. We also analyze the stability and convergence of our methods and illustrate how the empirical performance is captured by the theoretical results which unveil the bias and variance trade-off.

Finally, we examine the problem of estimating and tracking non-stationary signals and outliers in noisy streaming data emanating from both static and time-varying graphs. In conjunction with adaptive algorithms and optimization methods, we incorporate the LGT approach and outlier estimation. Through simulations and theoretical performance analysis we demonstrate the efficacy of this LGT-based approach, which is scalable and suitable for handling large time-varying graphs.

# Acknowledgements

First and foremost, I would like to express my sincere gratitude to my adviser, Dr. Manohar N. Murthi, and my co-advisor, Dr. Kamal Premaratne, for their continuous support for my Ph.D. research. Their patience, motivation, immense knowledge, and guidance helped me in conducting research as well as writing this dissertation. I could not have hoped for better advisers and mentors for my Ph.D. study.

I would also like to thank my fellow doctoral students for their feedback, cooperation and for all the fun we have had together in the last five years.

Last but not the least, I would like to thank my family. Support and guidance from my parents, my husband and my brothers helped me to overcome many challenges in my life to reach this achievement.

May Zar Lin

*University of Miami*

*May 2019*

# Table of Contents

# List of Figures

# List of Tables

# CHAPTER 1

# Introduction

Making sense of large amounts of data is a core challenge of signal processing. Distributed adaptive estimation, e.g., Least Mean Squares (LMS) [1–4] and Recursive Least Square (RLS) [5–8] principles, offers a potential approach to tackling this challenge. Distributed estimation relies only on local information and collaboration with the neighbors to attain the centralized performance while reducing the complexity. Moreover, due to its scalability and robustness, distributed estimation has been greatly used in many fields, e.g., sensor networks [9], social networks [10], brain response modelling [11] and biological networks [12], etc.

When the signals being estimated can be interpreted as emanating from an underlying graph structure, then graph signal processing (GSP) methods offer additional tools for analysis and design [13–17]. The Graph Fourier Transform (GFT) has recently emerged as a promising method for processing graph-based signals [18–21]. Researchers have exploited GSP in many applications such as sensor networks [22], big data analysis [23], brian imaging [24,25], image processing [26–29], signal classification [30], denoising [31–34], signal reconstruction and sampling [35–39] and anomaly detection [40]. The fundamental tools of GSP such as graph sampling [15, 32, 37, 41–44],

graph filtering [45–47], graph frequency analysis [48, 49] and optimal graph filter design [45–47, 50–53] have been developed in recent years.

How does one design systems to estimate and track non-stationary signals emanating from graph/network structures, and buried in noise? How does one simultaneously estimate possible additional outliers signals that might be of interest, e.g., outliers injected maliciously into some of the graph nodes. What if the graph is not fixed, but is time-varying, e.g., the edge weights describing the graph Laplacian are changing? And how does one tackle these issues with computationally efficient and scalable solutions whose performance can be analytically characterized prior to deployment?

To address these challenges, researchers have proposed a number of solutions that contain various mixes of adaptation and learning, distributed computation, optimization, and graph theory. For example several variations of adaptive filters have been proposed for tracking non-stationary signals with noise and outliers, with distributed solutions proposed to handle either sensor network applications, or effect efficient parallel computations [3, 54–57]. When one can interpret the multiple streams as having emanated from a graph/network structure that captures similarities between streams, then additional methods such as graph-based total variation penalization can be incorporated into adaptation and learning methods [18, 58].

However, these approaches have limitations when one wants to scale to large numbers of streams (and hence large graphs), and time-varying graphs in which the relations between streams are changing, or nodes are added/removed to/from the graph. For example, GSP methods often rely upon eigen-decomposition which become computationally expensive for very large graphs, and for time-varying graphs. Clearly methods that can handle large time-varying graphs are welcome. What about

identifying the outliers? While robust PCA methods explicitly identify the outlier components in data , it is not clear how to efficiently update such models for streaming data.

## 1.1 Our Approaches and Contributions

We first examine adaptive estimation methods such as Least-Mean-Square (LMS) and Recursive Least-Square (RLS) within the framework of graph signal processing. Such methods are popular for learning and tracking non-stationary signals in sensor networks, e.g., weather stations exchanging data and tracking environmental conditions. When a graph Laplacian matrix captures the similarities of the signal being sensed across nodes, then Laplacian Regularized (LR) LMS and LR RLS can provide estimation and tracking performance improvements. A natural question is whether GSP can be utilized to obtain additional performance improvements. We affirmatively answer this question by first interpreting the centralized algorithm LR LMS within GSP, with the regularization step viewed as a certain graph filtering operation. Then we demonstrate how to design optimal graph filters for this task with the caveat that the practitioner needs to know a signal of interest's maximum possible energy at each graph frequency.

Furthermore, we are interested in solutions that can scale to large time-varying network. We re-examine the distributed estimation of non-stationary signals with a penalty term now based on a time-varying graph Laplacian. We decompose the centralized cost into a sum of local costs at each node $k$, with node $k$'s cost based on its local time-varying Laplacian. We introduce a Local Graph Transform (LGT) approach to designing local filters within the diffusion adaptation LMS and RLS

paradigm. This leads to a solution in which each node $k$ only requires local information collected from its neighbors, and no global knowledge, meaning that our new method can scale to large time-varying graphs. We provide simulation and analytical performance bounds that demonstrate that this purely distributed solution is better than a centralized solution.

For both the centralized and distributed cases, we study the analytical stability and convergence, obtaining the theoretical mean squared deviation which illuminates how our graph filter-endowed methods achieve performance gains by trading off bias and variance. Simulation results confirm the attainment of the theoretical performance measures, and also illustrate the performance advantage of our methods over existing methods.

We further examine the fundamental challenge of estimating and tracking both non-stationary signals defined over time-varying graphs and outliers. We assume that the non-stationary ground truth signal follows a random walk model in which the correlated random step is connected to the time-varying graph structure and a random subset of nodes is perturbed by outliers signals that must be estimated as well. To address this challenge, we incorporate the LGT filters into an optimization-based adaptation and learning method. We provide a theoretical performance analysis regarding both stability and mean squared deviation that provide useful bounds and guidance to the practitioner. Through simulations, we demonstrate the efficacy of this approach in estimating and tracking both non-stationary ground truth signals and outliers signals in noise.

## 1.2 Organization of This Dissertation Proposal

The remainder of this proposal is organized as follows. In Chapter 2, we review the LR LMS and provides global/centralized graph filters for LR LMS with performance analysis. We also provide the global/centralized graph filter endowed distributed diffusion adaptation implementation with performance analysis and simulation results.

In Chapter 3, we decompose the centralized Laplacian regularization LMS and RLS cost into the sum of local costs. We introduce for each node $k$, an LGT-based local filters and study the diffusion adaptation LMS and RLS strategies incorporating the LGT filters in the estimation of the graph based signal. We provide performance analysis and simulations that demonstrate that this purely distributed solution is better than distributed LMS and RLS with traditional penalty based methods.

Chapter 4 examine the fundamental challenge of estimating and tracking both non-stationary signals and outlier signals from noisy data from time-varying graphs. Through simulations, we demonstrate the efficacy of this approach in estimating and tracking both non-stationary ground truth signals and outlier signals in noise. Furthermore, we provide a theoretical performance analysis regarding both stability and mean squared deviation.

Chapter 5 explains the future direction of this research work followed by the conclusion of this dissertation.

# CHAPTER 2

# Graph Signal Estimation

## 2.1 Centralized Graph Signal Estimation

Consider a sensor network with $N$ nodes. At time $i$, node $k$ has the measurement data $\{y_k(i), u_k(i)\}$ such that

$$y_k(i) = u_k(i)x_k^o + e_k(i) \tag{2.1}$$

in which $x_k^o$ is the node $k$'s true signal and $e_k(i)$ is node $k$'s measurement noise. For simplicity, we consider $y_k(i)$ and $u_k(i)$ to be a scalars, though the extension to multivariate/vector data is straightfoward. Suppose we consider the streams simultaneously by defining stacked vectors $\{\mathbf{y}_i, U_i\}$ in which $\mathbf{y}_i = \{y_1(i); \ldots; y_N(i)\}_{N\times 1}$, $U_i = diag\{u_1(i); \ldots; u_N(i)\}_{N\times N}$. Then network aggregate measurement is

$$\mathbf{y}_i = U_i\mathbf{x}^o + \mathbf{e}_i \tag{2.2}$$

where $\mathbf{e}_i = \{e_1(i); \ldots; e_N(i)\}_{N\times 1}$ and $\mathbf{x}^o$ is the ground truth network signal, $\mathbf{x}^o = \{x_1^o; \ldots; x_N^o\}_{N\times 1}$. The goal of the network is to estimate $\mathbf{x}^o$. The network cost function becomes

$$J^{glob}(\mathbf{x}) = \mathbb{E}\|\mathbf{y}_i - U_i\mathbf{x}\|^2 \tag{2.3}$$

where $\mathbf{x}$ is the estimate of $\mathbf{x}^o$. Eq.(2.3) can be solved using straightforward Least Squares. LMS achieves the solution in an iterative manner for the centralized case and via pareto diffusion adaptation [59] in a distributed manner. These adaptive methods are more suitable for tracking non-stationary $\mathbf{x}_i^o$. Now let assume that the normalized graph Laplacian matrix describing the similarities among the ground truth signal at different nodes/streams is given by $L = I_N - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ where $D$ is a diagonal matrix, $[D]_{k,k} = \sum_{l=1}^{N} [W]_{k,l}$ and $W$ is a weighted adjacency matrix. We assume that the ground truth is a non-stationary signal that evolves according to a random walk model with the initial ground truth signal $\mathbf{x}_0^o$, and that the random step $\mathbf{q}_i$ is a random perturbation:

$$\mathbf{x}_i^o = \mathbf{x}_{i-1}^o + \mathbf{q}_i. \tag{2.4a}$$

Here $\mathbf{q}_i$ is a zero mean Gaussian random vector with covariance matrix $cL^\dagger$ where $c$ is a non negative constant. Such random walk models are useful for studying the theoretical tracking ability of learning and adaptation algorithms [1], [4]. In this case, this model captures the fact that the ground truth signal's evolution is connected to a graph. We define the convex Laplacian regularization function as,

$$h(\mathbf{x}_i^o) = \sum_{k=1}^{N} \sum_{\ell \in \mathcal{N}_k} \frac{[W]_{\ell,k}}{2} \left| \frac{x_k^o(i)}{\sqrt{d_k}} - \frac{x_\ell^o(i)}{\sqrt{d_\ell}} \right|^2, \forall i. \tag{2.5}$$

where $\mathcal{N}_k$ is a set of node $k$'s neighbors. Using (2.5), we modify the network cost function from (2.3) as follows,

$$J^{Rglob}(\mathbf{x}_i) = \mathbb{E}\|\mathbf{y}_i - U_i \mathbf{x}_i\|^2 + \beta h(\mathbf{x}_i) \tag{2.6}$$

in which $\beta$ is a non-negative regularization parameter. We add the Laplacian regularizer to the LMS cost in (2.3) to minimize the total variation across the net-

work, [18], [48], [31] and [32] etc. Using stochastic gradient methods, we get the solution for (2.6) as,

$$\boldsymbol{\psi}_i = \mathbf{x}_{i-1} + \mu U_i^*(\mathbf{y}_i - U_i\mathbf{x}_{i-1}) \tag{2.7a}$$

$$\mathbf{x}_i = (I_N - \beta L)\boldsymbol{\psi}_i \tag{2.7b}$$

where $\mu$ is a non-negative step-size, $\boldsymbol{\psi}_i$ is an intermediate estimate of $\mathbf{x}_i^o$ and $\mathbf{x}_i$ is the estimate of $\mathbf{x}_i^o$ at time $i$. Using the Graph Fourier Transform, we interpret (2.7b) as a graph filter. Then we will show how to design an optimal graph filter to achieve better performance.

### 2.1.1 Graph Frequency Response Design and Global Graph Filtering

The Laplacian of the unidirected network can be written as $L = V\Lambda V^T$ where $V$ is the eigenvector matrix of $L$ and $V^T$ is denoted as a Graph Fourier Transform (GFT) matrix and $\Lambda$ is the diagonal eigenvalue matrix with $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$. In GFT domain, the eigenvalues are denoted as GFT frequencies with $\lambda_1$ and $\lambda_N$ as the lowest and highest frequencies, respectively. From [60] and [57], the regularization parameter $\beta$ can be designed adaptively by minimizing the following cost function

$$\min_{\beta} \mathbb{E}\|\mathbf{x}_i^o - (I_N - \beta L)\boldsymbol{\psi}_i\|^2. \tag{2.8}$$

From (2.8), we can write

$$\begin{aligned}\mathbb{E}\|\mathbf{x}_i^o - (I_N - \beta L)\boldsymbol{\psi}_i\|^2 &= \mathbb{E}\|\mathbf{x}_i^o - \left(I_N - \beta V\Lambda V^T\right)\boldsymbol{\psi}_i\|^2 \\ &= \mathbb{E}\|V^T\mathbf{x}_i^o - \left(I_N - \beta\Lambda\right)V^T\boldsymbol{\psi}_i\|^2. \end{aligned} \tag{2.9}$$

Let $\mathbf{f}_i^o$ and $\mathbf{\Psi}_i$ be the GFT frequency representation of $\mathbf{x}_i^o$ and $\boldsymbol{\psi}_i$, respectively, i.e., $\mathbf{f}_i^o = V^T\mathbf{x}_i^o$ and $\mathbf{\Psi}_i = V^T\boldsymbol{\psi}_i$. We denote $\mathbf{f}_i^o(\lambda_n)$ and $\mathbf{\Psi}_i(\lambda_n)$ are the frequency representations of $\mathbf{x}_i^o$ and $\boldsymbol{\psi}_i$ at frequency $\lambda_n$. Hence, the cost function in (2.8) can be rewritten as

$$\min_{\beta} \sum_{n=1}^{N} \mathbb{E}|\mathbf{f}_i^o(\lambda_n) - (1-\beta\lambda_n)\mathbf{\Psi}_i(\lambda_n)|^2 \tag{2.10}$$

From [60] and [57], the solution to Eq. (2.10) is

$$\beta(i) = \max\left\{\frac{2(\sum_{n=1}^{N}\lambda_n\mathbf{\Psi}_i(\lambda_n)^2 - \eta)}{\sum_{n=1}^{N}\lambda_n^2\mathbf{\Psi}_i(\lambda_n)^2}, 0\right\} \tag{2.11}$$

where $\eta$ is a threshold for the regularization function. Now we modify (2.7b) as

$$\mathbf{x}_i = \underbrace{(I_N - G_i)}_{\text{Global graph filter.}} \boldsymbol{\psi}_i \tag{2.12}$$

in which $G_i$ is a graph operator with $G_i = V\Theta_i V^T$ in which $\Theta_i = diag\{\theta_1(i), \ldots, \theta_N(i)\}$. We denote $\theta_n(i)$ as the frequency response of the graph filter $G_i$ at the frequency $\lambda_n$. Since $\lambda_1 = 0$, we use $\theta_1(i) = 0$. Then (2.8) can be rewritten as

$$\min_{\theta_n} \mathbb{E}|\mathbf{f}_i^o(\lambda_n) - (1-\theta_n)\mathbf{\Psi}_i(\lambda_n)|^2,\ n \in \{2, \ldots N\}. \tag{2.13}$$

Let $\eta_n$ be the maximum energy $|\mathbf{f}_i^o(\lambda_n)|^2$ of $\mathbf{x}_i^o$, $\forall i$ at frequency $\lambda_n$. Assume the network has $\eta_n$ for $n \in \{2, \ldots N\}$ and for a given $\mathbf{\Psi}_i(\lambda_n)$, $\theta_n(i)$ can be calculated as

$$\theta_n(i) = \max\left\{1 - \frac{\eta_n}{\mathbf{\Psi}_i(\lambda_n)^2}, 0\right\} \leq \theta_{max} \tag{2.14}$$

in which $\theta_{max}$ is the maximum frequency response derived in Section 2.2.2.

**Theorem 1** *For a given intermediate estimates $\boldsymbol{\psi}_i$ of $\mathbf{x}_i^o$ at time $i > 0$, the graph filter frequency response design using* (2.14) *can achieve the estimator performance gain over calculating the regularization parameter $\beta(i)$ from* (2.11)*.*

$$\mathbb{E}\|\mathbf{x}_i^o - (I_N - G_i)\boldsymbol{\psi}_i\|^2 \leq \mathbb{E}\|\mathbf{x}_i^o - (I_N - \beta(i)L)\boldsymbol{\psi}_i\|^2 \tag{2.15}$$

*in which $\beta(i)$ and $G_i$ are calculated using* (2.11) *and* (2.14)*, respectively.*

**proof: 1** *From* (2.10)*, we can rewrite*

$$\sum_{n=1}^{N} \mathbb{E}\left|\mathbf{f}_i^o(\lambda_n) - (1-\beta\lambda_n)\mathbf{\Psi}_i(\lambda_n)\right|^2 = \sum_{n=1}^{N} \mathbb{E}\left|\mathbf{f}_i^o(\lambda_n) - \mathbf{\Psi}_i(\lambda_n)\right|^2 + \beta^2 \sum_{n=1}^{N} \lambda_n^2 \mathbb{E}\mathbf{\Psi}_i(\lambda_n)^2 + \beta \sum_{n=1}^{N} \lambda_n \mathbb{E}\left(\mathbf{f}_i^o(\lambda_n) - \mathbf{\Psi}_i(\lambda_n)\right)\mathbf{\Psi}_i(\lambda_n) \tag{2.16}$$

*and Eq.* (2.11) *calculates $\beta$ such that*

$$\beta(i) \sum_{n=1}^{N} \lambda_n^2 \mathbf{\Psi}_i(\lambda_n)^2 + \sum_{n=1}^{N} \lambda_n \mathbb{E}\left(\mathbf{f}_i^o(\lambda_n) - \mathbf{\Psi}_i(\lambda_n)\right)\mathbf{\Psi}_i(\lambda_n) \leq 0. \tag{2.17}$$

*However, from Eq.* (2.17)*, it can be seen that for some $n$,*

$$\beta(i)\lambda_n^2 \mathbf{\Psi}_i(\lambda_n)^2 + \lambda_n \mathbb{E}\left(\mathbf{f}_i^o(\lambda_n) - \mathbf{\Psi}_i(\lambda_n)\right)\mathbf{\Psi}_i(\lambda_n) \nleq 0. \tag{2.18}$$

*However, from* (2.13) *and* (2.14)*, we have*

$$\theta_n(i)\mathbf{\Psi}_i(\lambda_n)^2 + \mathbb{E}\left(\mathbf{f}_i^o(\lambda_n) - \mathbf{\Psi}_i(\lambda_n)\right)\mathbf{\Psi}_i(\lambda_n) \leq 0, \ \ \forall n. \tag{2.19}$$

*Therefore, using* (2.17)*,* (2.18) *and* (2.19)*, it can be written as*

$$\sum_{n=1}^{N} \left(\theta_n(i)\mathbf{\Psi}_i(\lambda_n)^2 + \mathbb{E}\left(\mathbf{f}_i^o(\lambda_n) - \mathbf{\Psi}_i(\lambda_n)\right)\mathbf{\Psi}_i(\lambda_n)\right) \leq \beta(i) \sum_{n=1}^{N} \left(\lambda_n^2 \mathbf{\Psi}_i(\lambda_n)^2 + \lambda_n \mathbb{E}\left(\mathbf{f}_i^o(\lambda_n) - \mathbf{\Phi}_i(\lambda_n)\right)\mathbf{\Psi}_i(\lambda_n)\right). \tag{2.20}$$

*Therefore, we conclude that with the proposed the frequency response $\theta_n(i)$ using* (2.14)*, the estimator can achieve a better performance than using the existing method* (2.11)*. We complete the proof for Theorem 1.* □

We denote (2.7) using the frequency response design from (2.14) as the Global Graph Filtering GGF algorithm which is given in Algorithm 1.

**Algorithm 1** Global Graph Filtering (GGF)

Initialize $\mathbf{x}_{-1} = 0$. The network has the eigenvector $V$ of $L$ and has $\bar{\eta} = \{\eta_1, \ldots, \eta_N\}$.
**For** $i \geq 0$ **do**

1. The network has $\{\mathbf{y}_i, U_i\}$.
2. Perform $\boldsymbol{\psi}_i = \mathbf{x}_{i-1} + \mu U_i^* \left(\mathbf{y}_i - U_i \mathbf{x}_{i-1}\right)$.
3. Find the filter responses. $\boldsymbol{\Psi}_i = V^T \boldsymbol{\psi}_i$.
   **For** $n = 2$ to $N$ **do**

   $$\theta_n(i) = \max\left\{1 - \tfrac{\eta_n}{\boldsymbol{\Psi}_i(\lambda_n)^2}, 0\right\} \leq \theta_{max}.$$

   **End.**
4. $\mathbf{x}_i = (I_N - G_i)\boldsymbol{\psi}_i$ where $G_i = V\Theta_i V^T$ and $\Theta_i = diag\{0, \theta_2(i), \ldots, \theta_N(i)\}$.

**End.**

## 2.2 Performance of Global Graph Filtering

### 2.2.1 Stability and Mean Convergence

Before we analyze the mean and the mean square performance of the GGF algorithm, we examine the steady state behavior of the network estimator $\mathbf{x}_i$. Using (2.7a) and (2.12), we write the GGF algorithm in a recursion as

$$\begin{aligned}\mathbf{x}_i &= \left(I_N - G_i\right)\left(I_N - \mu U_i^* U_i\right)\mathbf{x}_{i-1} + \mu\left(I_N - G_i\right)U_i^*\mathbf{y}_i \\ &= \left(I_N - G_i\right)\left(I_N - \mu U_i^* U_i\right)\mathbf{x}_{i-1} + \mu\left(I_N - G_i\right)U_i^* U_i \mathbf{x}_i^o + \mu\left(I_N - G_i\right)U_i^*\mathbf{e}_i \end{aligned} \tag{2.21}$$

For simplicity, we assume $\mathbb{E}G_i = G$. Let $R_{u,i} = \mathbb{E}U_i^* U_i$ and $\mathbb{E}R_{u,i} = R_u$. Taking expectation to the both side of (2.21), we arrive at,

$$\mathbb{E}\mathbf{x}_i = \left(I_N - G\right)\left(I_N - \mu R_u\right)\mathbb{E}\mathbf{x}_{i-1} + \mu\left(I_N - G\right)R_u \mathbb{E}\mathbf{x}_i^o \tag{2.22}$$

We limit

$$\| \left(I_N - G_i\right)\left(I_N - \mu R_u\right) \| < 1, \forall i \tag{2.23}$$

and select the maximum filter response such that

$$\theta_{max} < \frac{1 + \rho\left(I_N - \mu R_u\right)}{\rho\left(I_N - \mu R_u\right)} \tag{2.24}$$

where $\rho(.)$ is the spectral radius of a matrix. From (2.4), we have $\mathbb{E}\mathbf{q}_i = 0$ and $\mathbb{E}\mathbf{x}_i^o = \mathbb{E}\mathbf{x}_{i-1}^o$. Let $\bar{\mathbf{x}}^o$ is the mean of the non-stationary true network signal where $\mathbb{E}\mathbf{x}_i^o = \bar{\mathbf{x}}^o$ for all $i$. Then using (2.23), we obtain the convergence in (2.22) as follows,

$$\begin{aligned}
\mathbb{E}\mathbf{x}_\infty &= \lim_{i\to\infty} \mathbb{E}\mathbf{x}_i \\
&= \bar{\mathbf{x}}^o - \underbrace{\left(I_N - \left(I_N - G\right)\left(I_N - \mu R_u\right)\right)^{-1} G\bar{\mathbf{x}}^o}_{\text{True signal rejected by GGF.}}. && (2.25) \\
&< \infty. && (2.26)
\end{aligned}$$

Observe that the estimator $\mathbf{x}_i$ converges to the weighted sum of the true signal filtered through the global graph filter. Therefore, when the graph filter response can represent the GFT frequency representation of the true signal $V^T\bar{\mathbf{x}}^o$ then $\left(I_N - G\right)\bar{\mathbf{x}}^o \approx \bar{\mathbf{x}}^o$. For example, the energy of $V^T\bar{\mathbf{x}}^o$ lies only in the low frequencies and the graph filter we use is a the low pass graph filter then we can achieve

$$\mathbb{E}\mathbf{x}_\infty \approx \bar{\mathbf{x}}^o \tag{2.27}$$

and the estimator becomes asymptotic unbiased.

### 2.2.2 Error Recursion and Bias

From previous analysis, the GGF algorithm some bias. Here we will study the steady state behavior of the network estimation error. Introduce the network's estimation error, $\tilde{\mathbf{x}} = \mathbf{x}_i^o - \mathbf{x}_i$. Using the expression in (2.21), the estimation error can be

expressed in a recursion form,

$$\begin{aligned}\tilde{\mathbf{x}}_i &= \mathbf{x}_i^o - (I_N - G_i)(I_N - \mu R_{u,i})\,\mathbf{x}_{i-1} - \mu (I_N - G_i) R_{u,i}\mathbf{x}_i^o - \mu (I_N - G_i) U_i^T \mathbf{e}_i \\ &= (I_N - G_i)(I_N - \mu R_{u,i})\,\tilde{\mathbf{x}}_{i-1} - (I_N - G_i)(I_N - \mu R_{u,i})\,\mathbf{x}_{i-1}^o \\ &\quad + (I_N - \mu (I_N - G_i) R_{u,i})\,\mathbf{x}_i^o - \mu (I_N - G_i) U_i^T \mathbf{e}_i \\ &= (I_N - G_i)(I_N - \mu R_{u,i})\,\tilde{\mathbf{x}}_{i-1} + G_i \mathbf{x}_i^o + (I_N - G_i)(I_N - \mu R_{u,i})\,\mathbf{q}_i - \mu (I_N - G_i) U_i^T \mathbf{e}_i \end{aligned} \tag{2.28}$$

Taking expectation to the both side (2.28), the expected error recursion becomes

$$\mathbb{E}\tilde{\mathbf{x}}_i = (I_N - G)(I_N - \mu R_u)\,\mathbb{E}\tilde{\mathbf{x}}_{i-1} + G\bar{\mathbf{x}}^o. \tag{2.29}$$

With the stability condition in (2.23), the expected estimation error of the GGF algorithm can be written as,

$$\begin{aligned}\mathbb{E}\tilde{\mathbf{x}}_\infty &= \lim_{i\to\infty} \mathbb{E}\tilde{\mathbf{x}}_i \\ &\approx \underbrace{\left(I_N - (I_N - G)(I_N - \mu R_u)\right)^{-1} G\bar{\mathbf{x}}^o}_{\text{Weighted sum of true signal rejected by GGF.}} < \infty. \end{aligned} \tag{2.30}$$

Note that the bias of the estimator is the weighted sum of the true signal components rejected by the global graph filter. Therefore, one can select an appropriate graph filter which represents the true underlying structure of the graph signal, thereby, reducing the estimator's bias.

### 2.2.3 Mean-Square Stability

Now, we study the steady state mean square behaviour of the GGF algorithm. Let $\|\tilde{\mathbf{x}}_i\|^2$ be the squared deviation of the network's estimation error at time $i$. From

(2.28), the squared deviation can be expressed as

$$\mathbb{E}\|\tilde{\mathbf{x}}_i\|^2 = \mathbb{E}\| \left(I_N - G_i\right)\left(I_N - \mu R_{u,i}\right)\tilde{\mathbf{x}}_{i-1}\|^2 + 2\mathbb{E}\ tr\left(\left(I_N - G_i\right)\left(I_N - \mu R_{u,i}\right)\tilde{\mathbf{x}}_{i-1}\mathbf{x}_i^{o^T} G_i\right)$$
$$+ \mathbb{E}\|G_i\mathbf{x}_i^o\|^2 + \mu^2\mathbb{E}\| \left(I_N - G_i\right) U_i^T\mathbf{e}_i\|^2 + \mathbb{E}\| \left(I_N - G_i\right)\left(I_N - \mu R_{u,i}\right)\mathbf{q}_i\|^2 \quad (2.31)$$

where $tr(.)$ is the trace of a matrix. Let $B = \left(I_N - G\right)\left(I_N - \mu R_u\right)$. From (2.29), we find the square of the expected error as,

$$\|\mathbb{E}\tilde{\mathbf{x}}_i\|^2 = tr\left(B\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)^T B^T + G\bar{\mathbf{x}}^o\bar{\mathbf{x}}^{o^T} G^T + 2B\mathbb{E}\tilde{\mathbf{x}}_{i-1}\bar{\mathbf{x}}^{o^T} G^T\right). \quad (2.32)$$

Let $\mathbb{E}\mathbf{e}_i\mathbf{e}_i^T = \Sigma$ and $\mathbb{E}\mathbf{q}_i\mathbf{q}_i^T = c\ L^\dagger$. By comparing (2.32) with (2.31), we get

$$\mathbb{E}\|\tilde{x}_i\|^2 = tr\Bigg(B\mathbb{E}\left(\tilde{\mathbf{x}}_{i-1}\tilde{\mathbf{x}}_{i-1}^T\right)B^T + \left(\mathbb{E}\tilde{\mathbf{x}}_i\right)\left(\mathbb{E}\tilde{\mathbf{x}}_i\right)^T - B\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)^T B^T + c\ BL^\dagger B^T$$
$$+ \mu^2\left(I_N - G\right)R_u\Sigma\left(I_N - G\right)^T + \mathbb{E}\left(G_i\left(\mathbf{x}_i^o - \bar{\mathbf{x}}^o\right)\left(\mathbf{x}_i^{o^T} - \bar{\mathbf{x}}^o\right)^T G_i\right)\Bigg). \quad (2.33)$$

From (2.4), the true signal $\mathbf{x}_i^o$ can be written as,

$$\mathbf{x}_i^o = \mathbf{x}_0^o + \sum_{j=0}^{i} q_j. \quad (2.34)$$

We assume $\mathbb{E}\left(G_i\mathbf{q}_j\right) = 0$ for $i \neq j$. Therefore,

$$E\left(G_i\left(\mathbf{x}_i^o - \bar{\mathbf{x}}^o\right)\left(\mathbf{x}_i^{o^T} - \bar{\mathbf{x}}^o\right)^T G_i\right) = G(\mathbb{E}\left(\mathbf{q}_i\mathbf{q}_i^T\right)G. \quad (2.35)$$

Then, we conclude the mean square error in the steady state as,

$$\mathbb{E}\|\tilde{\mathbf{x}}_\infty\|^2 \approx \underbrace{\|\mathbb{E}\tilde{\mathbf{x}}_\infty\|^2}_{\text{bias squared.}} + \underbrace{\mu^2\sum_{j=0}^{\infty} tr\Big(B^j(I_N - G)R_u\Sigma(I_N - G)B^{j^T}\Big)}_{\text{error due to the measurement noise.}}$$
$$\underbrace{+c\sum_{j=0}^{\infty} tr\Big(B^j(I - \mu(I - G)R_u)L^\dagger(I - \mu(I - G)R_u)^T B^{j^T}\Big)}_{\text{error due to the non-stationary data model.}}. \quad (2.36)$$

Let $MSD^g$ be the network mean-sqaure deviation of the GGF algorithm defined as,

$$MSD^g = \frac{1}{N}\mathbb{E}\|\tilde{\mathbf{x}}_\infty\|^2 < \infty. \quad (2.37)$$

## 2.3 Distributed Global Graph Filtering

In this section, we are interested in to find the distributed implementation of the centralized graph signal estimation in (2.7). We add the graph filtering step at each node in the diffusion adaptation algorithm [3]. Each node $k$ has the $k^{th}$ row of the eigenvector matrix $V$ and with extra message passing, each node $k$ performs GGF in a distributed manner. At time $i$, each node $k$ has $\{\mathbf{y}_{k,i}, U_{k,i}\}$ where $\mathbf{y}_{k,i}$ is an $N \times 1$ vector with $y_k(i)$ at the $k^{th}$ element and zero elsewhere. One can see that the network measurement signal $\mathbf{y}_i = \sum_{k=1}^{N} \mathbf{y}_{k,i}$. Let $U_{k,i}$ be an $N \times N$ matrix with $u_k(i)$ at the $k^{th}$ diagonal element and zero elsewhere, $U_i = \sum_{k=1}^{N} U_{k,i}$. Node $k$ is trying to learn the entire graph signal $\mathbf{x}_i^o$. Let $\mathbf{x}_{k,i}$ be node $k$'s estimate of $\mathbf{x}_i^o$ and $\mathbf{f}_{k,i} = V^T \mathbf{x}_{k,i}$. Rewrite (2.6) as the sum of local costs in a GFT domain as

$$\min_{\mathbf{f}_{k,i}, \forall k} \sum_{l \in \mathcal{N}_k} a_{\ell k}^{(1)} \mathbb{E} \|\mathbf{y}_{\ell i} - U_{\ell i} V \mathbf{f}_{k,i}\|^2 + \sum_{\ell \in \mathcal{N}_k \setminus \{k\}} a_{\ell k}^{(2)} \|\mathbf{f}_{k,i} - \mathbf{f}_{\ell i}\|^2 + \mathbf{f}_{k,i}^T \Theta \mathbf{f}_{k,i} \tag{2.38}$$

in which $a_{\ell k}^{(1)}$ and $a_{\ell k}^{(2)}$ are non-negative coefficients. Let $V_k$ be an $N \times N$ matrix whose the $k^{th}$ row contains the $k^{th}$ row of $V$ and zero elsewhere, $V = \sum_{k=1}^{N} V_k$. We denote $V_k$ as node $k$'s GFT coefficient matrix. As $U_{k,i}$ has a non-zero element only at the $k^{th}$ diagonal, one can easily verify that $U_{k,i} V = U_{k,i} V_k$. We find the solution for (2.38) as,

$$\boldsymbol{\Psi}_{k,i} = \mathbf{f}_{k,i-1} + \mu_k \sum_{\ell \in \mathcal{N}_k} a_{\ell k}^{(1)} V_\ell^T U_{\ell i}^T (\mathbf{y}_{\ell i} - U_{\ell i} V_\ell \mathbf{f}_{k,i-1}) \tag{2.39a}$$

$$\boldsymbol{\Phi}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}^{(2)} \boldsymbol{\Psi}_{\ell i} \tag{2.39b}$$

$$\mathbf{f}_{k,i} = (I_N - \Theta_{k,i}) \boldsymbol{\Phi}_{k,i} \tag{2.39c}$$

in which $\mu_k$ is a non-negative step size and $\boldsymbol{\Psi}_{k,i}$ and $\boldsymbol{\Phi}_{k,i}$ are node $k$'s intermediate estimates of $\mathbf{f}_i^o$. Each node designs its frequency response $\Theta_{k,i}$ using (2.14). Now we

need to transform $\mathbf{f}_{k,i}$ in the frequency domain back to $\mathbf{x}_{k,i}$ to a vertex domain. Since each node $k$ does not have complete knowledge of the GFT transform matrix $V^T$, instead each node $k$ has the $k^{th}$ row of $V$, using the message passing, each node $k$ find $\mathbf{x}_{k,i}$. Let us introduce the cost function

$$\min_{\mathbf{x}_{k,i},\forall k} \mathbb{E}\|\mathbf{x}_{k,i} - \sum_{\ell\in\mathcal{N}_k} a_{\ell k}^{(3)} V_\ell \mathbf{f}_{k,i}\|^2 + \sum_{\ell\in\mathcal{N}_k\setminus\{k\}} a_{\ell k}^{(4)} \|\mathbf{x}_{k,i} - \mathbf{x}_{\ell i}\|^2 \tag{2.40}$$

where $a_{\ell k}^{(3)}$ and $a_{\ell k}^{(4)}$ are non-negative coefficients. We get

$$\boldsymbol{\psi}_{k,i} = \mathbf{x}_{\ell i-1} - M_k\Big(\mathbf{x}_{\ell i-1} - \sum_{\ell\in\mathcal{N}_k} a_{\ell k}^{(3)} V_\ell \mathbf{f}_{\ell i}\Big) \tag{2.41a}$$

$$\mathbf{x}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}^{(4)} \boldsymbol{\psi}_{\ell i}. \tag{2.41b}$$

Since $\sum_{\ell\in\mathcal{N}_k} V_\ell \mathbf{f}_{\ell i}$ returns non-zero elements at the $\ell \in \mathcal{N}_k$ elements, we use a diagonal matrix, $M_k$ with a non-negative step size $\tilde{\mu}_k$ at the $\ell \in \mathcal{N}_k$ diagonal elements and zeros elsewhere. $\boldsymbol{\psi}_{k,i}$ is node $k$'s intermediate estimate. In (2.41a), each node $k$ exchanges $f_{k,i}$ and $V_k$ with its neighbors and update $\boldsymbol{\psi}_{k,i}$. Then, exchange $\boldsymbol{\psi}_{k,i}$ with its neighbors to update $x_{k,i}$. We denote (2.39) and (2.41) as the Distributed Global Graph Filtering (DGGF) detailed in Algorithm 2.

## 2.4 Performance of Distributed Global Graph Filtering

### 2.4.1 Stability and Bias

In this section, we analyze the stability conditions for the DGGF algorithm. Let $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ are weighted adjacency matrices, where $[A]_{l,k} = a_{l,k}$, $\sum_{l=1}^{N} a_{l,k} = 1$, $a_{l,k} = 0, \forall l \notin N_k$. Let $\mathcal{Q}_i = diag\{\Theta_{1,i}; \ldots; \Theta_{N,i}\}_{N^2\times N^2}$,

$\mathcal{S}_i = diag\{V_1^T U_{1,i}^T; \ldots; V_N^T U_{N,i}^T\}_{N^2\times N^2}$,

**Algorithm 2** Distributed Global Graph Filtering (DGGF) Alg.

Initialize $\mathbf{x}_{k,-1} = 0$ and $\mathbf{f}_{k,-1} = 0$ for all $k \in \mathcal{N}$. Each node $k$ has a set of neighbors $\mathcal{N}_k$ and access to $V_\ell$ $\ell \in \mathcal{N}_k$. Each node has $\eta_n$ for $n \in \{2, \ldots, N\}$.
**For** $i \geq 0$ and $k = 1$ to $N$ **do**

1. Each node $k$ has $\{\mathbf{y}_{k,i}, U_{k,i}\}$.
2. Exchange $\{\mathbf{y}_{k,i}, U_{k,i}\}$ with the neighbors.
3. $\mathbf{\Psi}_{k,i} = \mathbf{f}_{k,i-1} + \mu_k \sum_{\ell \in \mathcal{N}_k} a_{\ell,k}^{(1)} V_\ell^T U_{\ell i}^T \big(\mathbf{y}_{\ell,i} - U_{\ell,i} V_\ell \mathbf{f}_{k,i-1}\big)$.
4. Exchange $\mathbf{\Psi}_{k,i}$ with the neighbors.
5. $\mathbf{\Phi}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}^{(2)} \mathbf{\Psi}_{\ell,i}$.
6. Find the filter responses.

   **For** $n = 2$ to $N$ **do**

   $\theta_{k,n}(i) = \max\left\{1 - \frac{1\eta_n}{\mathbf{\Phi}_{k,i}(\lambda_n)^2}, 0\right\} \leq \theta_{k,max}$.

   **End.**
7. $\mathbf{f}_{k,i} = (I_N - \Theta_{k,i})\mathbf{\Phi}_{k,i}$, $\Theta_{k,i} = \{0, \theta_{k,2}(i), \ldots, \theta_{k,N}(i)\}$.
8. $\boldsymbol{\varrho}_{k,i} = \mathbf{x}_{k,i-1} - \tilde{\mu}_k \Big(\mathbf{x}_{k,i-1} - \sum_{\ell \in \mathcal{N}_k} V_\ell \mathbf{f}_{k,i}\Big)$.
9. Exchange $\boldsymbol{\varrho}_{k,i}$ with the neighbors.
10. $\mathbf{x}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}^{(3)} \boldsymbol{\varrho}_{\ell i}$.

**End.**

$\widetilde{\mathcal{V}} = diag\{\sum_{l\epsilon\mathcal{N}_1} V_l; \ldots; \sum_{l\epsilon\mathcal{N}_N} V_l\}_{N^2\times N^2}$, $\mu = \mu_k$ for all $k$ and $\mathcal{M}_1 = \mu I_{N^2}$ .

$\mathcal{M}_2 = diag\{M_1; \ldots; M_N\}_{N^2\times N^2}$. Let $\tilde{\mathbf{f}}_i^d$ and $\tilde{\mathbf{x}}_i^d$ be the network estimation error of DGGF algorithm in the GFT domain and the vertex domain, respectively, at time $i$, i.e.,

$$\tilde{\mathbf{f}}_i^d = \{\mathbf{f}_i^o - \mathbf{f}_{1,i}; \ldots; \mathbf{f}_i^o - \mathbf{f}_{N,i}\}_{N^2\times 1}, \quad \tilde{\mathbf{x}}_i^d = \{\mathbf{x}_i^o - \mathbf{x}_{1,i}; \ldots; \mathbf{x}_i^o - \mathbf{x}_{N,i}\}_{N^2\times 1} \tag{2.42}$$

where $\mathbf{f}_i^o = V^T\mathbf{x}_i^o$. From Eq. (2.39) and (2.41), we can write

$$\begin{bmatrix}\tilde{\mathbf{f}}_i^d\\ \tilde{\mathbf{x}}_i^d\end{bmatrix} = \begin{bmatrix}(I_{N^2} - \mathcal{Q}_i)\,\mathcal{A}_2^T\,(I_{N^2} - \mathcal{M}_1\mathcal{S}_i\mathcal{S}_i^T) & \emptyset\\ \mathcal{M}_2\mathcal{A}_3^T\tilde{\mathcal{V}} & \mathcal{A}_4^T(I_{N^2} - \mathcal{M}_2)\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{f}}_{i-1}^d\\ \tilde{\mathbf{x}}_{i-1}^d\end{bmatrix} + \begin{bmatrix}\mathcal{Q}_i(\mathbf{1}_N\otimes\mathbf{f}_i^o)\\ \mathbf{0}_N\end{bmatrix}$$
$$+ \begin{bmatrix}(I_{N^2} - \mathcal{Q}_i)\mathcal{A}_2^T(I_{N^2} - \mathcal{M}_1\mathcal{S}_i\mathcal{S}_i^T) & (I_{N^2} - \mathcal{Q}_i)\mathcal{A}_2^T\mathcal{S}_i)\\ \emptyset & \emptyset\end{bmatrix}\begin{bmatrix}(\mathbf{1}_N\otimes\mathbf{q}_i)\\ (\mathbf{1}_N\otimes\mathbf{e}_i)\end{bmatrix} \tag{2.43}$$

where $\emptyset$ is an $N\times N$ all zero matrix, $\mathbf{0}_N$ and $\mathbf{1}_N$ are an $N\times N$ all-zero matrix, an $N\times 1$ all-zero vector and an $N\times 1$ all-one vector, respectively. Let $\mathbb{E}\mathcal{S}_i = \mathcal{S}_i$. Since $\mathbb{E}\mathbf{e}_i = \mathbf{0}_N$ and $\mathbb{E}\mathbf{q}_i = \mathbf{o}_N$ and $\mathbb{E}\mathbf{f}_i^o = \bar{\mathbf{f}}^o = V^T\bar{\mathbf{x}}^o$, the expected value of Eq. (2.43) becomes

$$\begin{bmatrix}\mathbb{E}\tilde{\mathbf{f}}_i^d\\ \mathbb{E}\tilde{\mathbf{x}}_i^d\end{bmatrix} = \begin{bmatrix}(I_{N^2} - \mathcal{Q}_i)\,\mathcal{A}_2^T\,(I_{N^2} - \mathcal{M}_1\mathcal{S}\mathcal{S}^T) & \emptyset\\ \mathcal{M}_2\mathcal{A}_3^T\tilde{\mathcal{V}} & \mathcal{A}_4^T(I_{N^2} - \mathcal{M}_2)\end{bmatrix}\begin{bmatrix}\mathbb{E}\tilde{\mathbf{f}}_{i-1}^d\\ \mathbb{E}\tilde{\mathbf{x}}_{i-1}^d\end{bmatrix} + \begin{bmatrix}\mathcal{Q}_i(\mathbf{1}_N\otimes\bar{\mathbf{f}}^o)\\ \mathbf{0}_N\end{bmatrix}. \tag{2.44}$$

In Eq. (2.44), we select

$$\|\,(I_{N^2} - \mathcal{Q}_i)\,\mathcal{A}_2^T\left(I_{N^2} - \mathcal{M}_1\mathcal{S}\mathcal{S}^T\right)\| < 1; \quad \|\mathcal{M}_2\mathcal{A}_3^T\tilde{\mathcal{V}}\| < 1; \quad \|\mathcal{A}_4^T(I_{N^2} - \mathcal{M}_2)\| < 1,\ \forall i. \tag{2.45}$$

and hence, we have Eq.(2.44) converges in the steady state. From (2.44), we conclude that,

$$\begin{aligned}\mathbb{E}\tilde{\mathbf{f}}_{\infty}^{d} &= \lim_{i\to\infty}\mathbb{E}\tilde{\mathbf{f}}_{i}^{d} \\ &\approx \sum_{j=0}^{\infty}\left(\prod_{t=j+1}^{\infty}\left(\left(I_{N^2}-\mathcal{Q}_t\right)\mathcal{A}_2^T\left(I_{N^2}-\mathcal{M}_1\mathcal{S}\mathcal{S}^T\right)\right)\mathcal{Q}_j\left(\mathbf{1}_N\otimes\bar{\mathbf{f}}^o\right)\right) < \infty. \qquad (2.46)\end{aligned}$$

$$\begin{aligned}\mathbb{E}\tilde{\mathbf{x}}_{\infty}^{d} &= \lim_{i\to\infty}\mathbb{E}\tilde{\mathbf{x}}_{i}^{d} \\ &\approx \left(I_{N^2}-\mathcal{A}_4^T\left(I_{N^2}-\mathcal{M}_2\right)\right)^{-1}\mathcal{M}_2\mathcal{A}_3^T\tilde{\mathcal{V}}\ \ \mathbb{E}\tilde{\mathbf{f}}_{\infty}^{d} < \infty. \qquad (2.47)\end{aligned}$$

### 2.4.2 Mean-Square Stability

Let $\mathbb{B}_i = \left(I_{N^2}-\mathcal{Q}_i\right)\mathcal{A}_2^T\left(I_{N^2}-\mathcal{M}_1\mathcal{S}\mathcal{S}^T\right)$. From Eq. (2.43), we obtain the steady state mean-squared deviation of the DGGF algorithm as follows;

$$\begin{aligned}\mathbb{E}\|\tilde{\mathbf{f}}_{\infty}^{d}\|^2 &= \lim_{i\to\infty}\mathbb{E}\|\tilde{\mathbf{f}}_{i}^{d}\|^2 \\ &= \underbrace{\|\mathbb{E}\tilde{\mathbf{f}}_{\infty}\|^2}_{\text{bias squared.}} + \underbrace{c\sum_{j=0}^{\infty}tr\left(\prod_{t=j+1}^{\infty}\mathbb{B}_t\left(\left(\mathbb{B}_j+\mathcal{Q}_j\right)\left(\mathbf{1}_N\mathbf{1}_N^T\otimes L_j^{\dagger}\right)\left(\mathbb{B}_j+\mathcal{Q}_j\right)^T\right)\prod_{t=j+1}^{i}\mathbb{B}_t^T\right)}_{\text{error due to the non-stationary data model.}} \\ &+ \underbrace{\sum_{j=0}^{\infty}tr\left(\prod_{t=j+1}^{\infty}\mathbb{B}_t\left(I_{N^2}-\mathcal{Q}_j\right)\mathcal{A}_2^T\mathcal{S}\left(\mathbf{1}_N\mathbf{1}_N^T\otimes\Sigma\right)\mathcal{S}^T\mathcal{A}_2\left(I_{N^2}-\mathcal{Q}_j\right)\prod_{t=j+1}^{i}\mathbb{B}_t^T\right)}_{\text{error due to the measurement noise.}} \qquad (2.48)\end{aligned}$$

Using the above expression, we can express the network mean-square deviation in steady state, $MSD^d$ as,

$$MSD^d = \frac{1}{N^2}\mathbb{E}\|\tilde{\mathbf{f}}_{\infty}^{d}\|^2 < \infty. \qquad (2.49)$$

## 2.5 Experiment

To evaluate the performance of the proposed algorithm, we consider an undirected network with 30 nodes randomly distributed over a plane. The network Laplacian $L$

is formed from the weighted Euclidean distances between the nodes. We generate the initial true network signal $\mathbf{x}_0^o$ is drawn from a Gaussian distribution with $\mathcal{N}(10, L^\dagger)$. We use the non-stationary data model in (2.4) with $\mathbf{q}_i$ as a zero mean Gaussian with the variance of $0.001L^\dagger$. We set $\mu_k = 0.7$, $\tilde{\mu}_k = 1$ for all $k$, $\mu = \frac{\mu_k}{N}$, $\theta_{max} = 1$. $a_{l,k}^{(1)} = \delta_{l,k}$ where $\delta$ is the Kronecker delta function. $a_{l,k}^{(2)} = a_{l,k}^{(3)} = \frac{1}{N_k}$. Each node $k$ has a maximum 6 neighbors within the radius of 0.3. We assume $U_i = I_N$ and at each time $i$, the network observes $\mathbf{y}_i = \mathbf{x}_i^o + \mathbf{e}_i$. We compare the MSD results of our methods; Global Graph Filtering Algorithm (GGF) in Alg.1 and Distributed Global Graph Filtering Algorithm (DGGF) in Alg.2 with the traditional LMS and Laplacian Regularized LMS with the regularized parameter design from [60] and [57] denoted as LR-LMS. The simulation results are averaged over 100 runs. Figure 2.1 (a) and (b) show the network topology and the MSD comparisons between the mentioned algorithms. Note that our methods; GGF and DGGF achieve the theoretical MSD values in (2.37) and (2.49), respectively. As we can see the convergence rate of the DGGF algorithm is slower than the global algorithms. However, after an efficient information exchange, the DGGF can be viewed as $N$ parallel graph filters that are less sensitive to noise and hence, its MSD value outperforms the centralized algorithms.

## 2.6 Conclusion

In this chapter, we examine Regularized LMS in both centralized and distributed settings, and use GSP notions to design the improved adaptive algorithms. Simulation results show the benefits of the design and the achievement of analytical performance values. In the following chapters, we focus on graph signal estimation on time-varying graphs.

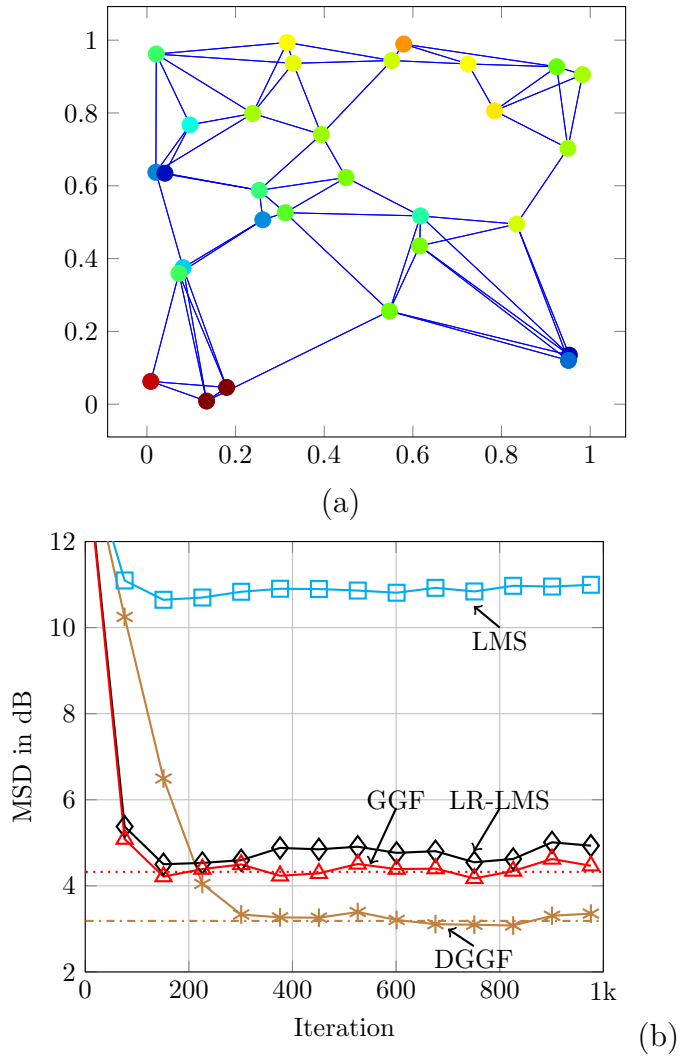



Figure 2.1: (a) Network Topology. The nodes' color represent the distribution of the true graph signal. (b) MSD comparison on traditional Least-Mean-Square (LMS), Laplacian-Regularized LMS with traditional regularized parameter design from [60] and [57] (LR-LMS), Global Graph Filtering (GGF in Alg.1) and Distributed Global Graph Filtering (DGGF in Alg.2). The dashed and dotted lines represent the corresponding theoretical MSD values calculated using Eq. (2.37) and (2.49), respectively. ($\mu = 0.7$, $SNR = -15.25dB$ and the graph signal are generated using (2.4) and $\mathbf{q}_i$ as a zero mean Gaussian random vector with the covariance matrix of $0.001\mathcal{L}^{\dagger}$.

# CHAPTER 3

# Graph Signal Estimation and Local Graph

In previous chapter, we examined such a Laplacian Regularized LMS (LR-LMS) scenario within the GSP framework, connecting the penalty term to graph filtering. However, when the graph is time varying, repeatedly computing the eigenbasis of a large graph is computationally expensive. We are interested in solutions that can scale to large numbers of nodes.

We start by re-examining the distributed estimation of non-stationary signals with a penalty term now based on a time-varying graph Laplacian. We decompose the centralized cost into a sum of local costs at each node $k$, with node $k$'s cost based on its local time-varying Laplacian. We introduce a Local Graph Transform (LGT) approach to designing local filters within the diffusion adaptation paradigm. This leads to a solution in which each node $k$ only requires local information collected from its neighbors, and no global knowledge, meaning that our new method can scale to large time-varying graphs.

## 3.1 Signal Estimation in Local Graph with LMS Strategy

First, let us rewrite the global regularization function (2.5) in a local form such that $h(\mathbf{x}_i^o) = \sum_{k=1}^{N} h_k(\mathbf{x}_i^o)$. Each $h_k$ is a local node $k$ penalty function given by

$$h_k(\mathbf{x}_i^o) = \sum_{\ell \in \mathcal{N}_k} \frac{[W]_{\ell,k}}{2} \left| \frac{x_k^o(i)}{\sqrt{d_k}} - \frac{x_\ell^o(i)}{\sqrt{d_\ell}} \right|^2 \tag{3.1}$$

in which $d_k = [D]_{k,k}$ and $w_{\ell,k} = [W]_{\ell,k}$ and $\mathcal{N}_k$ is a set of node $k$'s neighbors. We

**Algorithm 3** Local qausi-Laplacian from Normalized Global Laplacian

Node $k$ knows its neighbors global ID.
**For** $i > 0$ and $k = 1$ to $N$ **do**
1) Each node $k$ knows $w_{\ell,k}$ for all $l \in \mathcal{N}_k$.
2) Calculate $d_k = \sum_{\ell \in \mathcal{N}_k} w_{\ell,k}$.
3) Form local weighted adjacency matrix, $[A_k^l]_{\ell,k} = [A_k^l]_{k,\ell} = \frac{w_{\ell,k}}{2}$.
4) Find the local weighted degree matrix, $D_k^l = diag\left(A_k^l \mathbf{1}_N\right)$.
5) Each node $k$ exchanges $d_k$ with neighbors.
6) Form $[D_k]_{\ell,\ell} = d_\ell$ for $\ell \in \mathcal{N}_k$.
7) Form an $N \times N$ local qausi-Laplacian $L_k = D_k^{-\frac{1}{2}} \left(D_k^l - A_k^l\right) D_k^{-\frac{1}{2}}$.
**End**

introduce the local Laplacian $L_k$ at node $k$, an $N \times N$ solely from the vantage point of node $k$, i.e., a star network. Each node $k$ constructs its Local Laplacian $L_k$ using Algorithm 3. Figure 3.1 has the example of forming local graphs from the global Laplacian. We re-write each node $k$'s local penalty from (3.1) using its local Laplcain $L_k$ as,

$$h_k(\mathbf{x}_i^o) = \mathbf{x}_i^{o^T} L_k \mathbf{x}_i^o. \tag{3.2}$$

One can verify that the expression in Eq.(3.1) and (3.2) are equivalent. When we consider time-varying network, Eq.(3.1) becomes

$$h_{k,i}(\mathbf{x}_i^o) = \mathbf{x}_i^{o^T} L_{k,i} \mathbf{x}_i^o \tag{3.3}$$

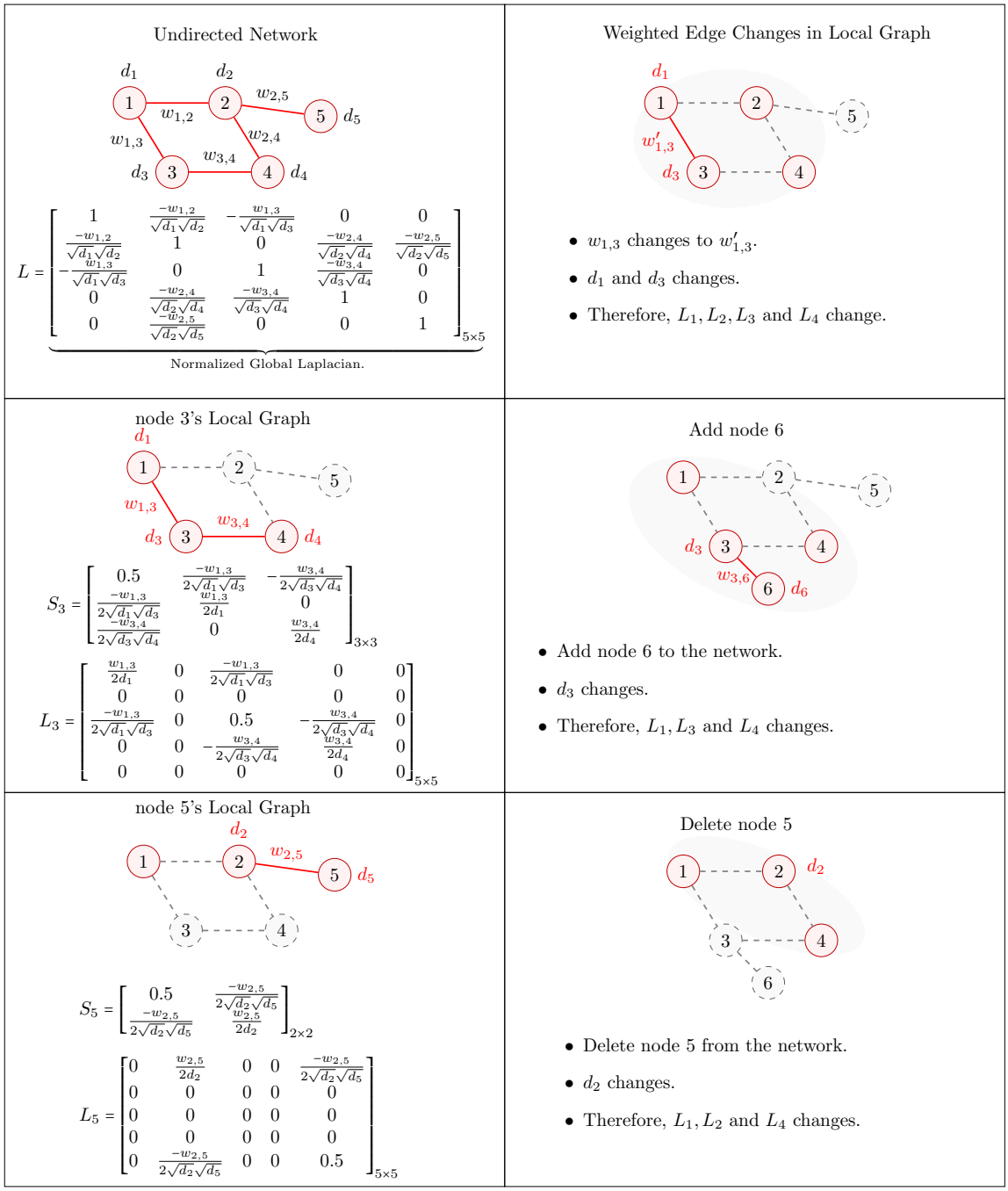



Figure 3.1: Example of local star graph and local Laplacian from global normalized Laplacian.

and the network Laplacian at time $i$, $L_i = \sum_{k=1}^{N} L_{k,i}$ and $h_{k,i}(\mathbf{x}_i^o)$ is denoted as node $k$'s local penalty function at time $i$. At time $i$, each node $k$ has $\{\mathbf{y}_{k,i}, U_{k,i}\}$ where $\mathbf{y}_{k,i}$ is an $N \times 1$ vector with $y_k(i)$ at the $k^{th}$ element and zero elsewhere, $\mathbf{y}_i = \sum_{k=1}^{N} \mathbf{y}_{k,i}$. Let $U_{k,i}$ be an $N \times N$ matrix with $u_k(i)$ at the $k^{th}$ diagonal element and zero elsewhere, $U_i = \sum_{k=1}^{N} U_{k,i}$. Node $k$ is trying to learn the entire graph signal $\mathbf{x}_i^o$. We define node $k$'s local cost as

$$J_k^{dist}(\mathbf{x}_{k,i}) = \sum_{\ell \in \mathcal{N}_{k,i}} a_{\ell,k}^{(1)}(i) \mathbb{E}\|\mathbf{y}_{\ell,i} - U_{\ell,i}\mathbf{x}_{k,i}\|^2 + \sum_{\ell \in \mathcal{N}_{k,i} \setminus \{k\}} a_{\ell,k}^{(2)}(i) \|\mathbf{x}_{k,i} - \mathbf{x}_{\ell,i}\|^2 + \beta_k h_{k,i}(\mathbf{x}_{k,i}) \tag{3.4}$$

in which $\mathcal{N}_{k,i}$ is a set of node $k$'s neighbors at time $i$ and $N_{k,i} = |\mathcal{N}_{k,i}|$ and $\mathbf{x}_{k,i}$ is node $k$'s estimate of $\mathbf{x}_i^o$, $\beta_k$ is node $k$'s non-negative regularization parameter and $a_{\ell,k}^{(1)}(i)$ and $a_{\ell,k}^{(2)}(i)$ are non-negative coefficients where $\sum_{\ell \in \mathcal{N}_{k,i}} a_{\ell,k}(i) = 1,\ a_{\ell,k}(i) = 0,\ \forall \ell \notin \mathcal{N}_{k,i}$. Then, the global cost (2.6) is the sum of local costs in (3.4), i.e.,

$$J^{Rglob}(\mathbf{x}_i) = \sum_{k=1}^{N} J_k^{dist}(\mathbf{x}_{k,i}). \tag{3.5}$$

We write the solution for (3.4) in three steps; Adapting, Combining and Filtering (ACF). The steps can be altered.

$$\boldsymbol{\phi}_{k,i} = \mathbf{x}_{k,i-1} + \mu_k \sum_{\ell \in \mathcal{N}_{k,i}} a_{\ell,k}^{(1)}(i) U_{\ell,i}^T (\mathbf{y}_{\ell,i} - U_{\ell,i}\mathbf{x}_{k,i-1}) \tag{3.6a}$$

$$\boldsymbol{\varphi}_{k,i} = \sum_{\ell \in \mathcal{N}_{k,i}} a_{\ell,k}^{(2)}(i) \boldsymbol{\phi}_{\ell,i} \tag{3.6b}$$

$$\mathbf{x}_{k,i} = \underbrace{(I_N - G_{k,i})}_{\text{DLMS-LGT filter.}} \boldsymbol{\varphi}_{k,i}. \tag{3.6c}$$

Instead of using the GFT, we introduce a Local Graph Transform (LGT) for each node $k$ using its local Laplacian, $L_{k,i}$. In (3.6c), we denote $(I - G_{k,i})$ as a node $k$'s

DLMS-LGT filter, with the design of $G_{k,i} = \beta_k L_{k,i}$ reverting to the ordinary local penalty. The key is to design $G_{k,i}$ to improve performance.

### 3.1.1 Local Laplacian and Local Graph Transform

In Chapter 2, we interpreted a similar equation as a graph filtering step that can be greatly improved by a graph frequency design. However, with very large time-varying graphs, repeatedly computing global eigenbases is costly. Therefore, here we introduce a Local Graph Transform (LGT) approach with the LGT for each node $k$ based on its local Laplacian At time $i$, each node $k$ has its local Laplacian $L_{k,i}$ which has only $N_{k,i}$ non-zero rows, columns, and diagonal elements. Let $P_{k,i}$ be a $N \times N_k$ suitable permutation matrix of node $k$ at time $i$ and

$$L_{k,i} = P_{k,i} S_{k,i} P_{k,i}^T \tag{3.7}$$

where $S_{k,i}$ is a small-scale local Laplacian matrix of size $N_{k,i} \times N_{k,i}$ corresponding to a star network structure. The eigen decomposition of $S_{k,i}$ can be written as

$$S_{k,i} = V_{k,i} \Lambda_{k,i} V_{k,i}^T \tag{3.8}$$

in which $\Lambda_{k,i} = diag\{\lambda_{k,1}(i), \ldots \lambda_{k,N_k}(i)\}$ is ascending. We denote $V_{k,i}^T$ as an $N_{k,i} \times N_{k,i}$ Local Graph Transform matrix of node $k$ at time $i$ and $\lambda_{k,1}(i), \ldots, \lambda_{k,N_k}(i)$ are denoted as the LGT eigenvalues of node $k$. We define $G_{k,i}$ in (3.6c) as

$$G_{k,i} = P_{k,i} V_{k,i} \Theta_{k,i} V_{k,i}^T P_{k,i}^T \tag{3.9}$$

where $\Theta_{k,i} = diag\{\theta_{k,1}(i), \ldots, \theta_{k,N_k}(i)\}$ and $\theta_{k,n}(i)$ is denoted as an LGT filter response with respect to the $\lambda_{k,n}(i)$ eigenvalue. For an ordinary local penalty, $\theta_{k,n}(i) =$

$\beta_k \lambda_{k,n}(i)$. Using (3.9), we can rewrite (3.6c) as

$$\mathbf{x}_{k,i} = \left(I_N - P_{k,i} V_{k,i} \Theta_{k,i} V_{k,i}^T P_{k,i}^T\right) \boldsymbol{\varphi}_{k,i}. \tag{3.10}$$

From (3.10), each node $k$ needs to know the $N_{k,i}$ eigenvectors of its time-varying local Laplacian $L_{k,i}$. We assume each node $k$ knows the change from $L_{k,i-1}$ to $L_{k,i}$ for all $i$. Therefore, the eigendecompositions are for small scale square matrices of dimension $N_{k,i}$, and not of dimension $N$, meaning that the method scales to large scale time-varying graphs.

### 3.1.2 Local Graph Transform Filter Response Design

Similar to the GFT frequency response design in Section 2.1.1, node $k$ finds the LGT filter response by minimizing the following cost,

$$\min_{\theta_{k,i}} \mathbb{E} \left\| \mathbf{x}_i^o - \left(I_N - P_{k,i} V_{k,i} \Theta_{k,i} V_{k,i}^T P_{k,i}^T\right) \boldsymbol{\varphi}_{k,i} \right\|^2, \forall k. \tag{3.11}$$

Let $\mathbf{f}_{k,i}^o$ and $\boldsymbol{\Phi}_{k,i}$ be the representation of $\mathbf{x}_i^o$ and $\boldsymbol{\varphi}_{k,i}$ in node $k$'s LGT such that $\mathbf{f}_{k,i}^o = V_{k,i}^T P_{k,i}^T \mathbf{x}_i^o$ and $\boldsymbol{\Phi}_{k,i} = V_{k,i}^T P_{k,i}^T \boldsymbol{\varphi}_{k,i}$. We denote $\mathbf{f}_{k,i}^o(\lambda_{k,n})$ as the LGT representation of $\mathbf{x}_i^o$ at the $\lambda_{k,n}(i)$ eigenvalue. Rewrite (3.11) as

$$\min_{\theta_{k,n}(i)} \mathbb{E} |\mathbf{f}_{k,i}^o(\lambda_{k,n}) - (1 - w_{k,n}(i))) \boldsymbol{\Phi}_{k,i}(\lambda_{k,n})|^2. \tag{3.12}$$

Assume node $k$ has $\zeta_{k,n}$, the upper limit for the energy $|\mathbf{f}_{k,i}^o(\lambda_{k,n})|^2$ for all $n$. At time $i$, for a given $\boldsymbol{\Phi}_{k,i}$ node $k$ calculates its LGT filter response as, [60], [57],

$$\theta_{k,n}(i) = \max\left\{1 - \frac{\zeta_{k,n}}{\boldsymbol{\Phi}_{k,i}(\lambda_{k,n})^2}, 0\right\} \leq \theta_{max} \tag{3.13}$$

in which $\theta_{max}$ is the maximum LGT filter response derived in Section 3.2.2. We denote (3.6) with the filter response design from (3.13) as Local Graph Transform

**Algorithm 4** Distributed Least Mean Square Strategy with Local Graph Transform (DLMS-LGT)

Initialize $\mathbf{x}_{k,0} = 0, \forall k$ and node $k$ has $\zeta_{k,n}, n \in \{1, \dots N_{k,i}\}$.
**For** $i > 0$ and $k = 1$ to $N$ **do**

1. Each node finds the neighbors $\mathcal{N}_{k,i}$ and has $L_{k,i}$.
2. Each node $k$ has $\{\mathbf{y}_{k,i}, U_{k,i}\}$.
3. Exchange $\{\mathbf{y}_{k,i}, U_{k,i}\}$ with the neighbors.
4. $\boldsymbol{\phi}_{k,i} = \mathbf{x}_{k,i-1} + \mu_k \sum_{\ell \in \mathcal{N}_{k,i}} a_{\ell,k}^{(1)}(i) U_{\ell,i}^T (\mathbf{y}_{\ell,i} - U_{\ell,i} \mathbf{x}_{k,i-1})$
5. Exchange $\{\boldsymbol{\phi}_{k,i}\}$ with the neighbors.
6. $\boldsymbol{\varphi}_{k,i} = \sum_{\ell \in \mathcal{N}_{k,i}} a_{\ell,k}^{(2)}(i) \boldsymbol{\phi}_{\ell,i}$.
7. **If** $L_{k,i} \neq L_{k,i-1}$, Compute $V_{k,i}$.
8. Find the local filter response.

   **For** $n = 1$ to $N_{k,i}$ **do**

   $\theta_{k,n}(i) = \max\left\{1 - \frac{1\zeta_{k,n}}{\boldsymbol{\Phi}_{k,i}(\lambda_{k,n})^2}, 0\right\} \leq \theta_{max}$, where $\boldsymbol{\Phi}_{k,i} = V_{k,i}^T P_{k,i}^T \boldsymbol{\varphi}_{k,i}$

   **End.**

   $\Theta_{k,i} = diag\{\theta_{k,1}(i), \dots, \theta_{k,N_{k,i}}(i)\}$.
9. $\mathbf{x}_{k,i} = \left(I_N - P_{k,i} V_{k,i} \Theta_{k,i} V_{k,i}^T P_{k,i}^T\right) \boldsymbol{\varphi}_{k,i}$.

**End.**

Filtering which is given in Algorithm 4.

**Remarks:** With LGT, one gains the advantage of a transform domain-based filter for enforcing local signal smoothness with respect to a local graph within the learning and adaptation algorithms. The node $k$ LGT is based on a local $L_{k,i}$ eigendecomposition, which must be re-computed with each local Laplacian change (edge addition, deletion, or weight change). This would incur a local cost of $\mathcal{O}(N_k^3)$. If $\bar{N_k}$ denotes the average number of nearest neighbors, then the worst case LGT updating cost is $\mathcal{O}(N * \bar{N_k}^3)$, assuming every node has to perform updating, with a lower cost if the graph variations are more localized at each updating instance. Updating the GFT requires $\mathcal{O}(N^3)$ which is much higher, considering that $\bar{N_k} \ll N$. When a edge weight changes in the network, LGT can update the local eigen pair with the cost of $\mathcal{O}(\bar{N_k}^4)$. For example, $w_{\ell,k}$ changes in the network effects changes $d_\ell$ and $d_k$ which are used in node $\ell$ and node $k$'s neighbors local Laplacian. Therefore, we need to recompute the eigen decomposition of at most $2\bar{N_k}$ nodes' Laplacian. Also LGT can efficiently update any adding/removing the node from the network. For example, we add node $m$ to the network and node $m$ is connected to node $k$, therefore, $d_k$ changes and since $d_k$ is used in node $k$'s neighbors' Laplacian. Therefore, adding/removing a node to the graph will effect changes the node's neighbors and their neighbors. Hence, the computation cost becomes $\mathcal{O}(\bar{N_k}^5)$. The detail complexity comparison between the Global Laplacian update and the normalized and un-normlaized LGT update for time-varying networks is given Table 3.1.

Also note that at initialization, each node $k$ should have some application knowledge regarding $\zeta_{k,n}$, the maximum possible signal energy with respect to each eigenvalue index, though this curve need not be updated during runtime, as it is assumed that

| Time Varying Network | Centralized Laplacian | Normalized LGT | Un-normalized LGT |
|---|---|---|---|
| the eigen decomposition of the network | $\mathcal{O}(N^3)$ | $\mathcal{O}(N.\bar{N}_k^3)$ | $\mathcal{O}(N.\bar{N}_k^3)$ |
| 1 edge weight changes. | $\mathcal{O}(N^3)$ | $\mathcal{O}\left(\min\left(N.\bar{N}_k^3, \bar{N}_k^4\right)\right)$ | $\mathcal{O}\left(\bar{N}_k^3\right)$ |
| $m$ edge weights change. | $\mathcal{O}(N^3)$ | $\mathcal{O}\left(\min\left(N.\bar{N}_k^3, {}^* m\bar{N}_k^3\right)\right)$ | $\mathcal{O}\left(m\bar{N}_k^3\right)$ |
| 1 node added/ removed. | $\mathcal{O}(N^3)$ | $\mathcal{O}\left(\min\left(N.\bar{N}_k^3, \bar{N}_k^5\right)\right)$ | $\mathcal{O}\left(\min\left(N.\bar{N}_k^3, \bar{N}_k^4\right)\right)$ |

*$m$ is the total number of nodes effects by the edge weights change.

Table 3.1: Computational complexity comparison of the eigen computation between the centralized Laplacian , nomarlized LGT and un-normalized LGT in time varying network. Note that $\bar{N}_k$ is the average number of neighbors in the network.

the non-stationary signal is smooth with respect to the graph, even though the graph is time-varying.

## 3.2 Performance of DLMS-Local Graph Transform

### 3.2.1 Stability and Mean Convergence

To study the steady state behavior of the learning in the network using the proposed DLMS-LGT algorithm, let $\mathcal{X}_i = \{\mathbf{x}_{1,i}; \ldots, \mathbf{x}_{N,i}\}_{N^2\times 1}$,

$\mathcal{Y}_i = \{\mathbf{y}_{1,i}; \ldots, \mathbf{y}_{N,i}\}_{N^2\times 1}$,

$\mathcal{U}_i = diag\{U_{1,i}; \ldots, U_{N,i}\}_{N^2\times N^2}$,

$\mathcal{R}_i = diag\{U_{1,i}^T U_{1,i}; \ldots, U_{N,i}^T U_{N,i}\}_{N^2\times N^2}$,

$\mathcal{X}_i^o = \mathbf{1}_N \otimes \mathbf{x}_i^o$, $\boldsymbol{\epsilon}_i = \mathbf{1}_N \otimes \mathbf{e}_i$, $Q_i = \mathbf{1}_N \otimes \mathbf{q}_i$,

$\mathcal{A}_{1,i} = A_{1,i} \otimes I_N$ and $\mathcal{A}_{2,i} = A_{2,i} \otimes I_N$ where $\otimes$ is the Kronecker product of the matrices $A_{1,i}$ and $A_{2,i}$ are weighted adjacency matrices where $[A]_{l,k} = a_{l,k}$, $\sum_{l=1}^{N} a_{l,k} = 1$, $a_{l,k} =$

$0, \forall l \notin N_k$. Let $\mu = \mu_k$ for all $k$ and from (2.4), we have

$$\mathcal{Y}_i = \mathcal{U}_i \mathcal{X}_i^o + \boldsymbol{\epsilon}_i \tag{3.14}$$

$$\mathcal{G}_i = \underbrace{diag\{I_N - P_{1,i}V_{1,i}\Theta_{1,i}V_{1,i}^T P_{1,i}^T, \ldots, I_N - P_{N,i}V_{N,i}\Theta_{N,i}V_{N,i}^T P_{N,i}^T\}.\mathcal{A}_{2,i}^T}_{\text{Network DLMS-LGT filters.}} \tag{3.15}$$

Note that we refer $\mathcal{G}_i$ as the network DLMS-LGT filters in later analysis. Now we write Eq.(3.6) in a recursion form as,

$$\mathcal{X}_i = \mathcal{G}_i(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_{u,i})\mathcal{X}_{i-1} + \mu \mathcal{G}_i \mathcal{A}_{1,i}^T \mathcal{R}_{u,i} \mathcal{X}_i^o + \mu \mathcal{G}_i \mathcal{A}_{1,i}^T \mathcal{U}_i \boldsymbol{\epsilon}_i \tag{3.16}$$

where $\mathcal{I}$ is an $N^2 \times N^2$ identity matrix. Let $\mathbb{E}\mathcal{R}_{u,i} = \mathcal{R}_u$, $\mathbb{E}U_i = U_i$, $\bar{\mathcal{X}}^o = \mathbf{1}_N \otimes \bar{\mathbf{x}}^o$ and we have $\mathbb{E}Q_i = 0$, then, we find the expected value of the above recursion as

$$\begin{aligned} \mathbb{E}\mathcal{X}_i &= \mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_u\right) \mathbb{E}\mathcal{X}_{i-1} + \mu \mathcal{G}_i \mathcal{A}_{1,i}^T \mathcal{R}_u \bar{\mathcal{X}}^o \\ &= \underbrace{\mu \sum_{j=0}^{i} \left( \prod_{t=j+1}^{i-1} \mathcal{G}_t \left(\mathcal{I} - \mu \mathcal{A}_{1,t}^T \mathcal{R}_u\right) \right) \mathcal{G}_j \mathcal{A}_{1,j}^T \mathcal{R}_u \bar{\mathcal{X}}^o}_{\text{True signal accepted by the network DLMS-LGT filters.}} \end{aligned} \tag{3.17}$$

For stability of (3.17), we need

$$\| \mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_u\right) \| < 1, \forall i. \tag{3.18}$$

Therefore, we select $\mu$ and each node $k$'s filter response such that

$$0 \leq \theta_{max} < \frac{1 - \rho\left(1 + \mu \mathcal{R}_u\right)}{\rho\left(1 - \mu \mathcal{R}_u\right)} \tag{3.19}$$

then, we can conclude $\lim_{i \to \infty} \mathbb{E}\mathcal{X}_i < \infty$. From (3.17), notice that the LGT base graph signal estimation is a biased estimator and it converges to the weighted sum of the true signal after passing through the DLMS-LGT filters. In the next section, we will study the asymptotic convergence of the bias and the mean square behavior.

### 3.2.2 Error Recursion and Bias

Let $\tilde{\mathbf{x}}_{k,i}$ be the estimation error of node $k$ in the LGT algorithm where

$$\tilde{\mathbf{x}}_{k,i} = \mathbf{x}_i^o - \mathbf{x}_{k,i}. \tag{3.20}$$

and we define the network estimation error as $\tilde{\mathcal{X}}_i = \{\tilde{\mathbf{x}}_{1,i}; \ldots; \tilde{\mathbf{x}}_{N,i}\}$. Using the recursion in (3.16), we have

$$\tilde{\mathcal{X}}_i = \mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_{u,i}\right) \tilde{\mathcal{X}}_{i-1} + \left(\mathcal{I} - \mathcal{G}_i\right) \mathcal{X}_i^o + \mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_{u,i}\right) Q_i - \mu \mathcal{G}_i \mathcal{A}_{1,i}^T \mathcal{U}_i \boldsymbol{\epsilon}_i \tag{3.21}$$

With (3.21), we obtain the expected network estimation error as,

$$\mathbb{E}\tilde{\mathcal{X}}_i = \mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_u\right) \mathbb{E}\tilde{\mathcal{X}}_{i-1} + \left(\mathcal{I} - \mathcal{G}_i\right) \bar{\mathcal{X}}^o \tag{3.22}$$

Once again, with $\|\mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_u\right)\| < 1$, the steady state expected error is

$$\begin{aligned} \mathbb{E}\tilde{\mathcal{X}}_\infty &= \lim_{i \to \infty} \mathbb{E}\tilde{\mathcal{X}}_i \\ &= \underbrace{\sum_{j=0}^{\infty} \left( \prod_{t=j+1}^{\infty} \mathcal{G}_t \left( \left(\mathcal{I} - \mu \mathcal{A}_{1,t}^T \mathcal{R}_u\right) \right) \left(\mathcal{I} - \mathcal{G}_j\right) \bar{\mathcal{X}}^o. \right)}_{\text{True signal rejected by network DLMS-LGT filters.}} < \infty. \end{aligned} \tag{3.23}$$

Notice that the expected learning error is the weighted sum of any true signal rejected from the network DLMS-LGT filters.

### 3.2.3 Mean-Square Stability

Let us explore the mean squared behavior of the LGT based graph signal estimation. Let $\mathcal{B}_i = \mathcal{G}_i \left(\mathcal{I} - \mu \mathcal{A}_{1,i}^T \mathcal{R}_{u,i}\right)$. From (3.21), we can write the mean-squared of the estimation error as

$$\begin{aligned} \mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 = \mathbb{E}\|\mathcal{B}_i \tilde{\mathcal{X}}_{i-1}\|^2 + \mathbb{E}\|\left(\mathcal{I} - \mathcal{G}_i\right) \mathcal{X}_i^o\|^2 + \mathbb{E}\|\mathcal{B}_i Q_i\|^2 + \mu^2 \mathbb{E}\|\mathcal{G}_i \mathcal{A}_{1,i}^T \mathcal{U}_i \boldsymbol{\epsilon}_i\|^2 \\ + 2tr\left(\mathcal{B}_i \left(\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right) \bar{\mathcal{X}}^{o^T} \left(\mathcal{I} - \mathcal{G}_i\right)^T\right) \end{aligned} \tag{3.24}$$

From (3.22), the square of the expected estimation error is

$$\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)^T = \mathcal{B}_i\left(\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right)^T\mathcal{B}_i^T + \left(\mathcal{I}-\mathcal{G}_i\right)\bar{\mathcal{X}}^o\bar{\mathcal{X}}^{oT}\left(\mathcal{I}-\mathcal{G}_i\right)^T + 2tr\left(\mathcal{B}_i\left(\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right)\bar{\mathcal{X}}^{oT}\left(\mathcal{I}-\mathcal{G}_i\right)^T\right) \tag{3.25}$$

Comparing the two expressions from (3.25) and (3.26) and following from the mean square derivation of the GGF algorithm in Section 2.2.3, we arrive the steady state mean square error of the LGT method as

$$\begin{aligned}\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 &= \lim_{i\to\infty}\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 \\ &\approx \underbrace{\|\mathbb{E}\tilde{\mathcal{X}}_\infty\|^2}_{\text{bias squared}} + \underbrace{tr\left(\sum_{j=0}^{\infty}\left(\prod_{t=j+1}^{\infty}\mathcal{B}_t\right)\mathcal{G}_j\mathcal{A}_{1,j}^T\mathcal{R}_u\left(\mathbf{1}_N\mathbf{1}_N^T\otimes\Sigma\right)\mathcal{A}_{1,i}^T\mathcal{G}_j^T\left(\prod_{t=j+1}^{\infty}\mathcal{B}_t^T\right)\right)}_{\text{error due to the measurement noise.}} \\ &\quad + \underbrace{c\ tr\left(\sum_{j=0}^{\infty}\left(\prod_{t=j+1}^{\infty}\mathcal{B}_t\right)\left(\mathcal{I}-\mu\mathcal{G}_j\mathcal{A}_{1,j}^T\mathcal{R}_u\right)\left(\mathbf{1}_N\mathbf{1}_N^T\otimes L_j^\dagger\right)\left(\mathcal{I}-\mu\mathcal{G}_j\mathcal{A}_{1,j}^T\mathcal{R}_u\right)\left(\prod_{t=j+1}^{\infty}\mathcal{B}_t^T\right)\right)}_{\text{error due to the non-stationary data model.}}\end{aligned} \tag{3.26}$$

where $\mathbb{E}\left(\boldsymbol{\epsilon}_i\boldsymbol{\epsilon}_i^T\right) = cL_i^\dagger$ and $\mathbb{E}Q_iQ_i^T = \left(\mathbf{1}_N\mathbf{1}_N^T\otimes\mathbb{E}\mathbf{q}_i\mathbf{q}_i^T\right)$ and $\mathbb{E}\mathbf{q}_i\mathbf{q}_i^T = \Sigma$. Let $MSD^l$ be the mean square deviation of the network defined as,

$$MSD^l = \frac{1}{N^2}\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 < \infty. \tag{3.27}$$

Next, we compare the steady state mean square error of the LGT based DLMS method with the traditional diffusion unbiased DLMS. For the sake of simplicity, we assume the network is static and from (3.15), we set $\Omega_{k,i} = 0$ for all $i$ and $k$, thereby, removing the regularization and $\mathcal{G}_i = \mathcal{I}$. Let $\mathbf{B} = \left(\mathcal{I}-\mu\mathcal{A}_1^T\mathcal{R}\right)$. We get the mean squared error

of the traditional diffusion LMS as

$$\mathbb{E}\|\tilde{\mathcal{X}}_i^{lms}\|^2 = \underbrace{tr\left(\sum_{j=0}^{i}\mathbf{B}^{i-j}\mathcal{A}_1^T\mathcal{R}_u\left(\mathbf{1}_N\mathbf{1}_N^T\otimes\Sigma\right)\mathcal{A}_1^T\mathbf{B}^{(i-j)^T}\right)}_{\text{error due to the measurement noise.}} + \underbrace{c\ tr\left(\sum_{j=0}^{i}\mathbf{B}^{i-j+1}\left(\mathbf{1}_N\mathbf{1}_N^T\otimes L_j^\dagger\right)\mathbf{B}^{(i-j+1)^T}\right)}_{\text{error due to the non-stationary data model.}}. \tag{3.28}$$

Let assume the DLMS-LGT filters in (3.15) are static, $\mathcal{G} = \mathcal{G}_i$. Let $d_n(i)$ be the difference between the error components due to the measurement from (3.26) and (3.28) respectively, i.e.,

$$\begin{aligned}
d_n(i) &= tr\Bigg(\sum_{j=0}^{i}\Big(\mathcal{B}_t^{i-j}\mathcal{G}\mathcal{A}_1^T\mathcal{R}_u\left(\mathbf{1}_N\mathbf{1}_N^T\otimes\Sigma\right)\mathcal{A}_1^T\mathcal{G}^T\mathcal{B}_t^{(i-j)^T} \\
&\quad - \mathbf{B}^{i-j}\mathcal{A}_1^T\mathcal{R}_u\left(\mathbf{1}_N\mathbf{1}_N^T\otimes\Sigma\right)\mathcal{A}_1^T\mathbf{B}^{(i-j)^T}\Big)\Bigg) \\
&= tr\left(\sum_{j=0}^{i}\left(\mathcal{G}^T\mathcal{B}_t^{(i-j)^T}\mathcal{B}_t^{i-j}\mathcal{G} - \mathbf{B}^{(i-j)^T}\mathbf{B}^{i-j}\right)\mathcal{A}_1^T\mathcal{R}_u\left(\mathbf{1}_N\mathbf{1}_N^T\otimes\Sigma\right)\mathcal{A}_1^T\right) \\
&\le tr\left(\sum_{j=0}^{i}\left(\mathcal{G}^T\mathcal{B}_t^{(i-j)^T}\mathcal{B}_t^{i-j}\mathcal{G} - \mathbf{B}^{(i-j)^T}\mathbf{B}^{i-j}\right)\right) \\
&= \left(\sum_{j=0}^{i}\left(\left(\mathcal{G}\mathbf{B}^{i-j}\mathcal{G}\right)^T\left(\mathcal{G}\mathbf{B}^{i-j}\mathcal{G}\right) - \mathbf{B}^{(i-j)^T}\mathbf{B}^{i-j}\right)\right)
\end{aligned} \tag{3.29}$$

where $\mathcal{B} = \mathcal{G}\mathbf{B}$ and we can limit each node $k$'s filter response $w_{max}$ to obtain $\|\mathcal{G}\| \le 1$ and then achieve $d_n(i) \le 0$ for all $i$. Next, we compare the two error components due to the non-stationary data model from (3.26) and (3.28) as,

$$\begin{aligned}
d_{ns}(i) &= ctr\left(\sum_{j=0}^{i}\left(\mathcal{I}-\mu\mathcal{G}\mathcal{A}_1\mathcal{R}_u\right)^T\mathcal{B}^{j^T}\mathcal{B}^{i-j}\left(\mathcal{I}-\mu\mathcal{G}\mathcal{A}_1\mathcal{R}_u\right) - \mathbf{B}^{(i-j)^T}\mathbf{B}^{j}\right)\left(\mathbf{1}_N\mathbf{1}_N^T\otimes L_j^\dagger\right) \\
&\le tr\left(\sum_{j=0}^{i}\left(\mathcal{I}-\mu\mathcal{G}\mathcal{A}_1\mathcal{R}_u\right)^T\mathbf{B}^{(i-j)^T}\mathcal{G}^T\mathcal{G}\mathbf{B}^{i-j}\left(\mathcal{I}-\mu\mathcal{G}\mathcal{A}_1\mathcal{R}_u\right) - \mathbf{B}^{(i-j)^T}\mathbf{B}^{i-j}\right) \\
&\le 1, \quad \forall i.
\end{aligned} \tag{3.30}$$

Therefore, we conclude that by allowing biased estimator with non-zero network DLMS-LGT filters $\mathcal{G}_i$, we can achieve the lower errors due to the measurement noise and the non-stationary signal model than that of the traditional unbiased estimator.

## 3.3 Experiments

### 3.3.1 Experiment using Synthetic Data

Here, we reconsider the undirected network from Section 2.5. We set $\tilde{\mu}_k = 1$ for all $k$, $\mu = \frac{\mu_k}{N}$, $\mu_k = 0.8$. We set $a_{l,k}^{(1)} = a_{l,k}^{(4)} = \delta_{l,k}$ where $\delta$ is the Kronecker delta function. $a_{l,k}^{(2)} = a_{l,k}^{(3)} = a_{l,k}^{(5)} = \frac{1}{N_k}$. Each node $k$ has a maximum 6 neighbors within the radius of 0.3. We assume $U_i = I_N$ and at each time, the network observes $\mathbf{y}_i = \mathbf{x}_i^o + \mathbf{e}_i$ with an $SNR$ of $-5dB$. We compare the MSD results of the Distributed LMS-Local Graph Transform (DLMS-LGT) Alg. 4 with the traditional LMS and and the Global Graph Filtering (GGF) Alg.1 for both the slowly and rapidly varying graph signal model. Table 3.2 mentions the initials for the compared algorithms used in the figures. We

| Algorithm | Initials |
|---|---|
| Traditional Least-Mean-Square | LMS |
| Global Graph Filtering Alg.1 | GGF |
| Distributed Laplacian Reguarlized LMS | Dist.LR-LMS |
| Distributed LMS with Local Graph Transform Alg.4 | DLMS-LGT |

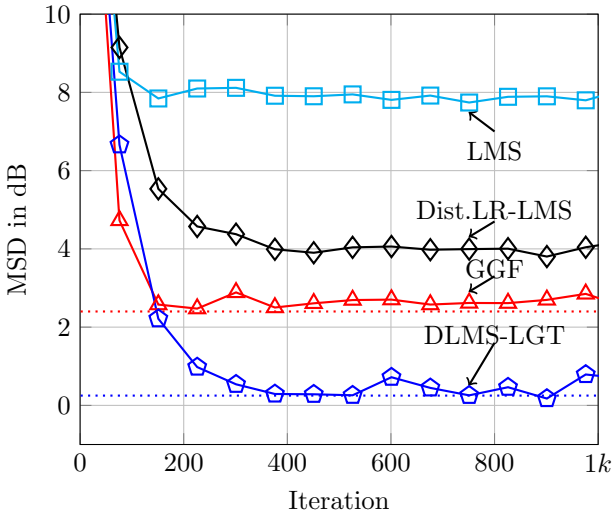
Table 3.2: Algorithms and Initials used in Figures

generate the slowly varying graph signal using (2.4) with the covariance matrix of $\mathbf{q}_i$ as $0.0025\mathcal{L}^\dagger$ and use $0.01\mathcal{L}^\dagger$ for rapidly varying graph signal. The simulation results are averaged over 100 runs. Figure 3.2 shows the MSD comparisons between the LMS, GGF and LGT algorithms. Note that the GGF and DLMS-LGT algorithms

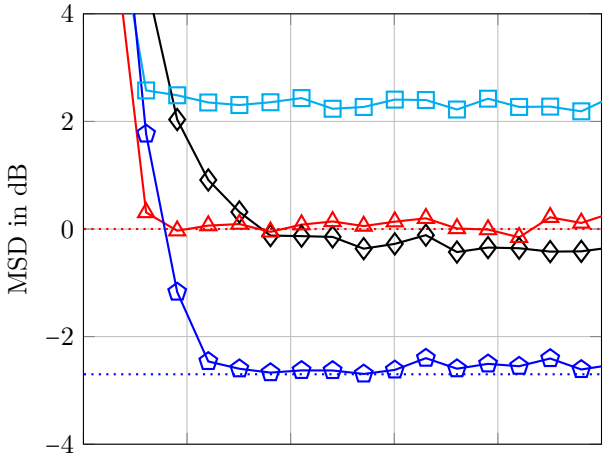
achieve the theoretical MSD values in (2.37) and (3.27). We also tested the proposed DLMS-LGT method in time varying graphs. At time $J_n$, the graph randomly changes 10% of the edges and $J_n$ are drawn from the Poission distribution with the average of 1 per 100 iteration. Figure 3.3 shows the MSD compariosn between the three algorithms with time varying graphs. As we learn that the proposed DLMS-LGT algorithm outperforms the global/centralized algorithms in all the test cases.

### 3.3.2 Experiment on Real-World Data

We tested the proposed algorithms; Global Graph Filtering (GGF) in Alg. 1, Distributed Global Graph Filtering (DGGF) in Alg. 2 and the DLMS-Local Graph Transform in Alg.4 on the average temperature recorded over 50 weather stations across the Florida, USA between January 1st, 2010 and December 31st, 2010 [61]. From [62], the weather station's temperature are correlated between the stations with similar elevation rather than their geographical distance. Therefore, we selected the weather stations across the Florida since the elevation difference across the Florida can be ignored. We randomly selected 50 weather stations out of 150 stations from the dataset, [61]. We form the network by connecting the nearest geographical distance stations with the maximum neighbors of 5. We set $\mu_k = 0.5$ and the other parameters are the same as in synthetic data simulation. We run 100 iteration per each day. The MSD results are averaged over 100 simulations. The station network topology and true temperature value are shown in Figure 3.4 (a) and (b), respectively. Figure 3.4 (c) shows the MSD comparisons between the traditional LMS, GGF, DGGF and DLMS-LGT. Each algorithm's initial is mentioned in Table 3.2 . Figure 3.5 shows the




(a)




(b)

Figure 3.2: MSD Comparison. The algorithm initials are mentioned in Table 3.2. The dotted and dashed lines are the calculated theoretical MSD values of the respective algorithms, (2.37) and (3.27). ($\mu = 0.7$, $SNR = -5dB$ and the graph signal are generated using (2.4) with $\mathbf{q}_i$ as a zero mean Gaussian random vector with the covariance matrix of (a): $0.01\mathcal{L}^{\dagger}$ and (b): $0.0025\mathcal{L}^{\dagger}$.

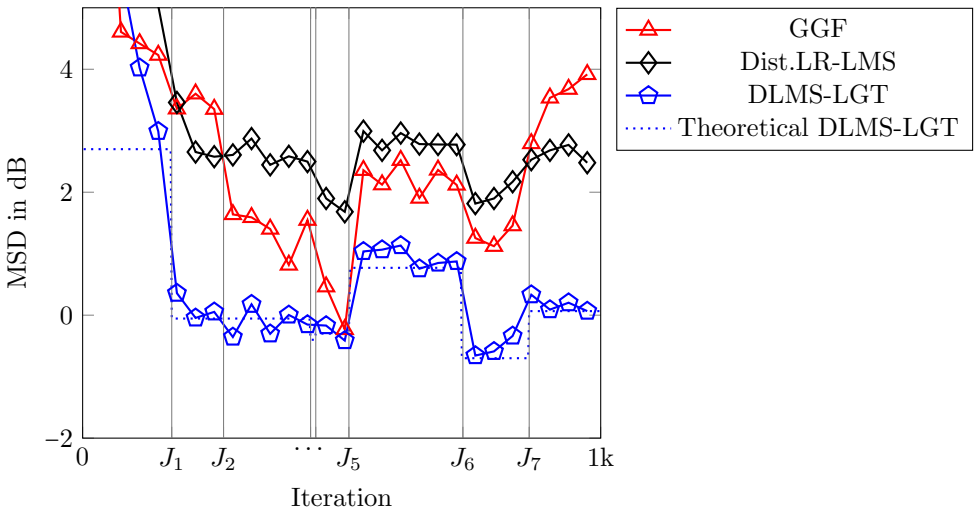



Figure 3.3: MSD Comparison on time varying graph. At $J_1, \ldots, J_7$, the graph changes 10% of the edges and $J_n$ are drawn from the Poission distribution with the average of 1 per 100 iterations.

tracking abilities of GGF and DLMS-LGT in comparison with the traditional LMS. We do not show the tracking abilities of DGGF algorithm since it is the distributed implementation of GGF and the results are similar to GGF. As we expected, the proposed algorithms, DLMS-LGT and GGF introduced some bias to the temperature value which however lead to the lower MSD value showed in Figure 3.4 (c).

## 3.4 Signal Estimation in Local Graph with LS Strategy

In this section, we introduce the graph signal estimation using the weighted least square strategy with the Local Graph Transform. Consider a network of $N$ nodes and the network has the measurement data $\mathbf{y}_i$ where $\mathbf{y}_i = \{y_1(i), \ldots, y_N(i)\}_{N\times 1}$ and $y_k(i)$ is each node $k$'s scalar measurement data. The network measurement data are

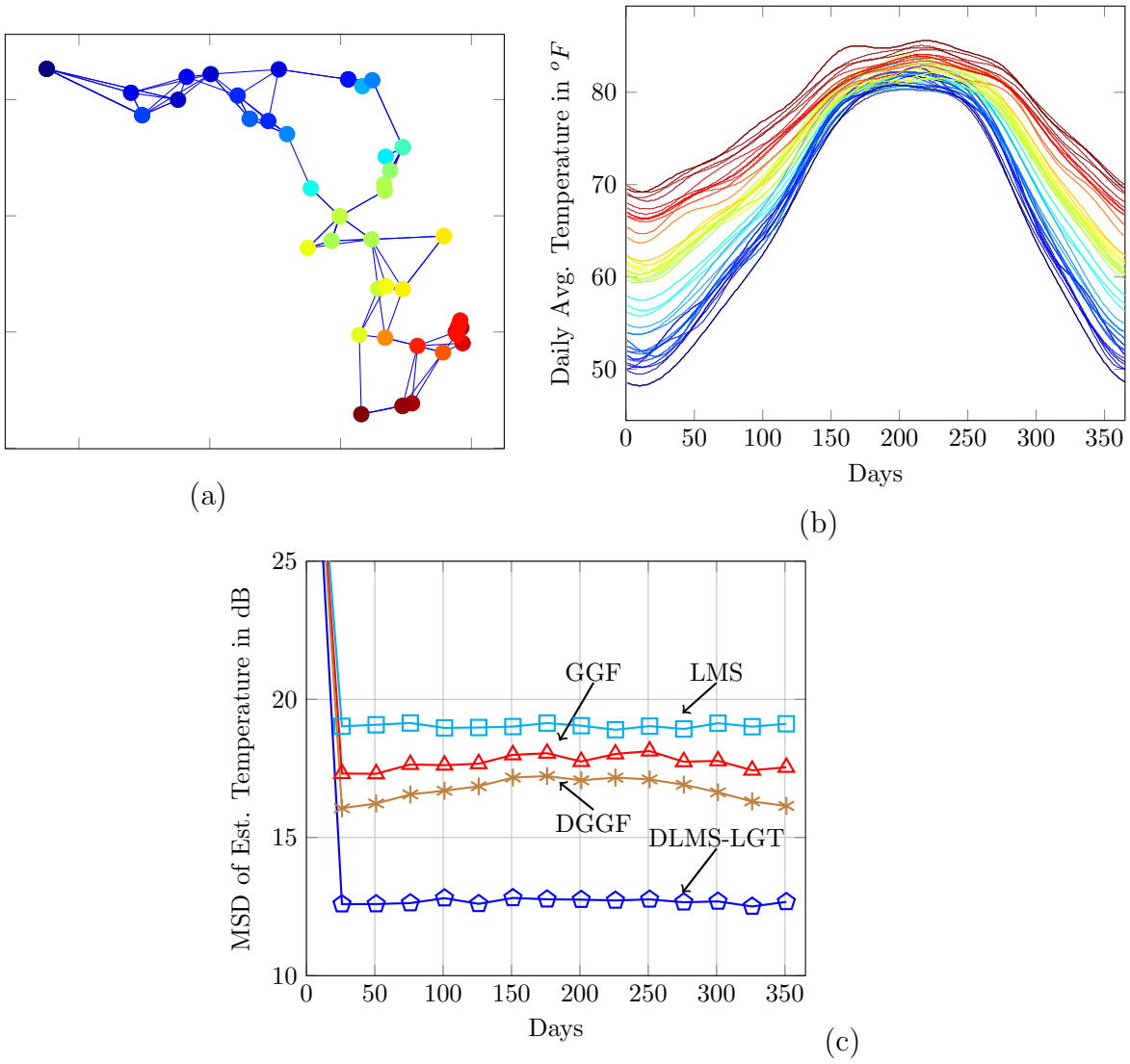



Figure 3.4: (a) Network topology of 50 Weather Stations across Florida, [61]. (b) Daily average temperature between January $1^{st}$, 2010 and December $31^{st}$, 2010. (c) MSD Comparisons on the average temperature value of 50 Weather Station across Florida, [61]. The algorithm initials are mentioned in Table 3.2. The true temperature values are corrupted with a zero-mean Gaussian noise and all the algorithms run 100 iteration per each day. $SNR = -2dB$ and $\mu_k = 0.5$.

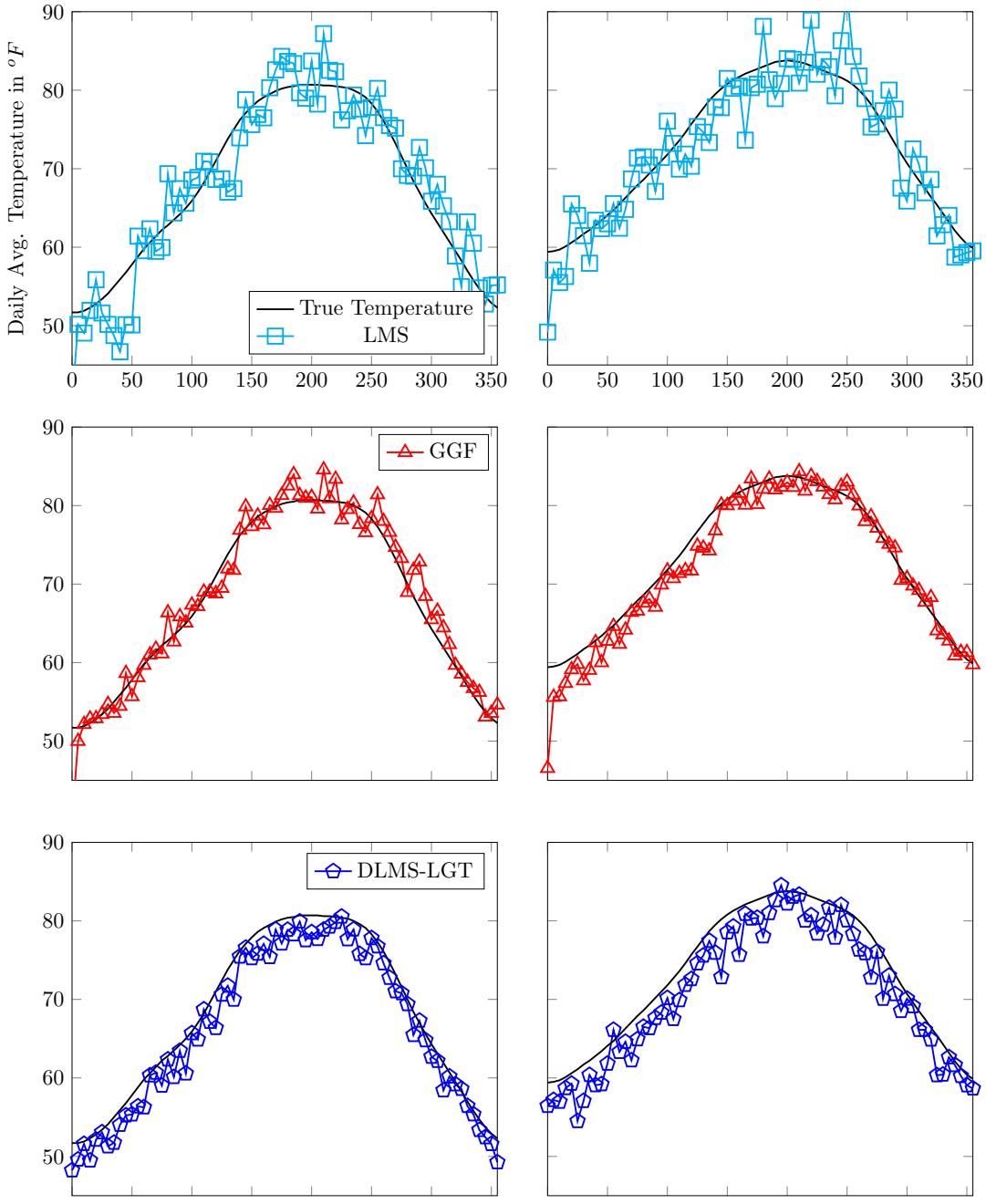



Figure 3.5: Tracking performance comparison of the average temperature recorded between Jan 1st, 2010 and Dec 31st, 2010 at Cross City, FL(Left) and Orlando, FL (Right). The Y-axis represents the average temperature value and the X-axis represents the days. The true temperature values are corrupted with a zero-mean Gaussian noise. $SNR = -2dB$ and $\mu_k = 0.5$.

defined as,

$$\mathbf{y}_i = \mathbf{x}_i^o + \mathbf{e}_i \tag{3.31}$$

where $\mathbf{x}_i^o$ is the $N \times 1$ the network's true signal and $\mathbf{e}_i$ is the additive noise with the size of $N \times 1$. Assume the network Laplacian $L$ captures the underlying structure of the true network signal $\mathbf{x}_i^o$. The network estimates $\mathbf{x}$ by minimizing the sum of squared differences weighted over the variations of the measurements:

$$J_i^g(\mathbf{x}) = \sum_{j=0}^{i} r^{i-j} \|\mathbf{y}_j - \mathbf{x}\|^2 + \beta \mathbf{x}^T L \mathbf{x} \tag{3.32}$$

where $0 < r < 1$ is a weighting parameter. We are interested in the distributed solution of (3.32). The goal of each node $k$ is to estimate its true signal and its neighbors' true signal. Let $\mathbf{y}_{k,i}$ be the $N_k \times 1$ which contains node $k$'s and its neighbors' true signalWe have $\mathbf{y}_i = \sum_{k=1}^{N} P_k \mathbf{y}_{k,i}$ where $P_k$ is an $N \times N_k$ permutation matrix. Let $\mathbf{x}_{i,i}^o$ be an $N_k \times 1$ vector contains node $k$ and its neighbors true signal, $\mathbf{x}_{k,i}^o = P_k^T \mathbf{x}_i^o$. The goal of each node $k$ is to estimate $\mathbf{x}_{k,i}^o$. Using the node $k$'s local Laplacian $S_k$ defined in Section 3.1, we write each node $k$'s local cost in (3.32) as,

$$J_i^{dist}(\mathbf{x}_k) = \sum_{j=0}^{i} r_k^{i-j} \sum_{\ell \in \mathcal{N}_k} \|P_k^T P_\ell \mathbf{y}_{\ell,j} - \mathbf{x}_k\|^2 + \beta_k \mathbf{x}_k^T L_k \mathbf{x}_k + \sum_{\ell \in N_k} a_{\ell,k} \|P_\ell \mathbf{x}_\ell - P_k \mathbf{x}_k\|^2 \tag{3.33}$$

and

$$J_i^g(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{k=1}^{N} J_i^{dist}(\mathbf{x}_k). \tag{3.34}$$

We write the solution for (3.33) in two steps; [5],

$$\boldsymbol{\phi}_{k,i} = \mathbf{x}_{k,i-1} + \frac{1}{1+r_k} \Big( \underbrace{\big(I + \ddot{G}_{k,i}\big)^{-1}}_{\text{DLS-LGT filter.}} \sum_{\ell \in \mathcal{N}_k} P_k^T P_\ell \mathbf{y}_{\ell,i} - \mathbf{x}_{k,i-1} \Big) \tag{3.35}$$

$$\mathbf{x}_{k,i} = \sum_{\ell \in N_k} a_{\ell,k} P_k^T P_\ell \boldsymbol{\phi}_{\ell,i} \tag{3.36}$$

where $I_k$ is an identity matrix of size $N_k \times N_k$. Similar to the DLMS-LGT filter introduced in Section 3.1.1, we denote $\left(I + \ddot{G}_{k,i}\right)^{-1}$ in (3.35) as a DLS-LGT graph filter where $\ddot{G}_k$ be node $k$'s graph operator with $\ddot{G}_{k,i} = V_k \ddot{\Theta}_{k,i} V_k^T$ where $\ddot{\Theta}_{k,i}$ is a diagonal matrix with $\ddot{\theta}_{k,n}$ for $n = \{1, \ldots, N_k\}$ denoted as node $k$'s local graph filter response and $V_k^T$ is node $k$'s LGT matrix. Let $\mathbf{f}_{k,i}^o$ and $\mathbf{\Psi}_{k,i}$ are the frequency representation of $\mathbf{x}_{k,i}^o$ and $\mathbf{y}_{k,i}$ respectively where $\mathbf{f}_{k,i}^o = V_k^T \mathbf{x}_{k,i}^o$ and $\mathbf{\Psi}_{k,i} = V_k^T \mathbf{y}_{k,i}$. Similar to DLMS-LGT filter response design in Section 3.1.2, each node $k$ designs its local graph filter response by minimizing the following cost as

$$\min_{\ddot{\theta}_{k,n}} \|\mathbf{f}_{k,i}^o(\lambda_{k,n}) - (1 + \ddot{\theta}_{k,n})^{-1} \mathbf{\Psi}_{k,i}(\lambda_{k,n})\|^2, \quad n \in \{1, \ldots, N_k\}, \forall k. \tag{3.37}$$

where $(\lambda_{k,n})$ is the node $k$'s LGT frequency and $\mathbf{f}_{k,i}^o(\lambda_{k,n})$ and $\mathbf{\Psi}_{k,i}(\lambda_{k,n})$ are the $n^{th}$ element of the vector $\mathbf{f}_{k,i}^o$ and $\mathbf{\Psi}_{k,i}$, respectively. From (3.37), we want to calculate the filter response such that

$$\begin{aligned} \|\mathbf{f}_{k,i}^o(\lambda_{k,n}) - (1 + \ddot{\theta}_{k,n})^{-1} \mathbf{\Psi}_{k,i}(\lambda_{k,n})\|^2 = \|\mathbf{f}_{k,i}^o(\lambda_{k,n}) - \mathbf{\Psi}_{k,i}(\lambda_{k,n})\|^2 + 2\left(\mathbf{f}_{k,i}^o(\lambda_{k,n}) - \mathbf{\Psi}_{k,i}(\lambda_{k,n})\right) \\ .\left(\mathbf{\Psi}_{k,i}(\lambda_{k,n}) - (1 + \ddot{\theta}_{k,n})^{-1} \mathbf{\Psi}_{k,i}(\lambda_{k,n})\right) \\ + \left(\mathbf{\Psi}_{k,i}(\lambda_{k,n}) - (1 + \ddot{\theta}_{k,n})^{-1} \mathbf{\Psi}_{k,i}(\lambda_{k,n})\right)^2 \end{aligned} \tag{3.38}$$

In (3.38), the first term $\|\mathbf{f}_{k,i}^o(\lambda_{k,n}) - \mathbf{\Psi}_{k,i}(\lambda_{k,n})\|^2$ is as the estimation error without the penalty. Therefore, we design the LGT filter response $\ddot{\theta}_{k,n}$ such that

$$\begin{aligned} 2\left(\mathbf{f}_{k,i}^o(\lambda_{k,n}) - \mathbf{\Psi}_{k,i}(\lambda_{k,n})\right)\left(\mathbf{\Psi}_{k,i}(\lambda_{k,n}) - (1 + \ddot{\theta}_{k,n})^{-1} \mathbf{\Psi}_{k,i}(\lambda_{k,n})\right) \\ + \left(\mathbf{\Psi}_{k,i}(\lambda_{k,n}) - (1 + \ddot{\theta}_{k,n})^{-1} \mathbf{\Psi}_{k,i}(\lambda_{k,n})\right)^2 \leq 0 \end{aligned} \tag{3.39}$$

For any real-valued convex function $g(.)$, from the definition of sub-gradient, we have

$$(x - y)^T \partial g(y) \leq g(x) - g(y) \tag{3.40}$$

and let assume $g(x) = x^2$. For the purpose of clarification, let $x = \mathbf{f}^o_{k,i}(\lambda_{k,n})$ and $y = \mathbf{\Psi}_{k,i}(\lambda_{k,n})$. Then, using (3.40), we rewrite (3.39) as

$$\begin{aligned}
\left(x^2 - y^2\right)\left(1 - \left(1 + \ddot{\theta}_{k,n}\right)^{-1}\right) + y^2 - 2\left(1 + \ddot{\theta}_{k,n}\right)^{-1} y^2 + \left(1 + \ddot{\theta}_{k,n}\right)^{-2} y^2 &\leq 0 \\
x^2\left(1 - \left(1 + \ddot{\theta}_{k,n}\right)^{-1}\right) - \left(1 + \ddot{\theta}_{k,n}\right)^{-1} y^2 + \left(1 + \ddot{\theta}_{k,n}\right)^{-2} y^2 &\leq 0 \\
x^2 - \left(1 + \ddot{\theta}_{k,n}\right)^{-1} y^2 &\leq 0 \\
\ddot{\theta}_{k,n} &\leq \frac{y^2}{x^2} - 1 \qquad (3.41)
\end{aligned}$$

Therefore, from the above inequality, we design each node $k$'s LGT filter response as follows;

$$\ddot{\theta}_{k,n}(i) = \max\left\{\frac{\mathbf{\Psi}_{k,i}(\lambda_{k,n})^2}{2\zeta_{k,n}} - \frac{1}{2}, 0\right\} \leq \ddot{\theta}_{max}. \qquad (3.42)$$

Hence, $\zeta_{k,n}$ is the maximum possible energy $|\mathbf{f}^o_{k,i}(\lambda_{k,n})|^2$, while $\ddot{\theta}_{max}$ is a maximum response. The complete distributed least square strategy with the LGT filter is given in Algorithm 5.

## 3.5 Performance of DLS-Local Graph Transform

### 3.5.1 Stability and Mean Convergence

Our aim is to study the steady state performance of the proposed algorithm. For the ease of the analysis, let us define the non stationary signal model for the network as, [1],

$$\mathbf{x}^o_i = a\mathbf{x}^o_{i-1} + \mathbf{q}_i \qquad (3.43)$$

**Algorithm 5** Distributed Least Square Strategy with Local Graph Transform (DLS-LGT)

Initialize $\mathbf{x}_{k,0} = \mathbf{y}_{k,0}$, and node $k$ has $\zeta_{k,n}, n \in \{1, \dots N_k\}$.
**For** $i > 0$ and $k = 1$ to $N$ **do**

1. Each node $k$ has $y_k(i)$.
2. Exchange $y_k(i)$ with the neighbors and node $k$ has $\mathbf{y}_{k,i}$.
3. Find the DRLS-LGT filter response,

   **For** $n = 1$ to $N_{k,i}$ **do**

   $$\ddot{\theta}_{k,n}(i) = \max\left\{\frac{\mathbf{\Psi}_{k,i}(\lambda_{k,n})^2}{2\zeta_{k,n}} - 1, 0\right\} \leq \ddot{\omega}_{max}.$$

   **End.**

   $\ddot{G}_{k,i} = V_k \ddot{\Theta}_{k,i} V_k^T$ where $\ddot{\Theta}_{k,i} = diag\{\ddot{\theta}_{k,1}(i), \dots, \ddot{\theta}_{k,N_k}(i)\}$.
4. $\boldsymbol{\phi}_{k,i} = \mathbf{x}_{k,i-1} + \frac{1}{1+r_k}\left(\left(I + \ddot{G}_{k,i}\right)^{-1}\mathbf{y}_{k,i} - \mathbf{x}_{k,i-1}\right)$
5. Exchange $\boldsymbol{\phi}_{k,i}$ with the neighbors.
6. $\mathbf{x}_{k,i} = \sum_{\ell \in N_k} a_{\ell,k} P_k^T P_\ell \boldsymbol{\phi}_{\ell,i}$.

**End.**

where $|a| < 1$ and $\mathbf{q}_i$ is some random perturbation drawn from $\mathbf{q}_i \sim \mathcal{N}(\bar{\mathbf{q}}, L^{\dagger})$.

$$\mathbb{E}\mathbf{x}_i^o = a\mathbb{E}\mathbf{x}_{i-1}^o + \bar{\mathbf{q}}. \tag{3.44}$$

Hence, in steady state, one can write $\bar{\mathbf{x}}^o = \lim_{i\to\infty} \mathbb{E}\mathbf{x}_i^o = \frac{1}{1-a}\bar{\mathbf{q}}$. From Eq.(3.35) and (3.36), we rewrite the DLS-LGT algorithm in a recursion form as

$$\mathbf{x}_{k,i} = \sum_{\ell\in N_k} \frac{a_{\ell,k}}{1+r_\ell} P_k^T P_\ell \left( \left(I_\ell + \ddot{G}_{\ell,i}\right)^{-1} \sum_{n\in\mathcal{N}_\ell} P_\ell^T P_n \mathbf{y}_{n,i} + r_\ell \mathbf{x}_{\ell,i-1} \right) \tag{3.45}$$

Let $\mathcal{P} = \{P_1, \ldots, P_N\}_{N\times\sum_{k=1}^N N_k}$,

$\mathcal{X}_i = \{\mathbf{x}_{1,i}, \ldots, \mathbf{x}_{N,i}\}_{\sum_{k=1}^N N_k\times 1}$, $\mathcal{X}_i^o = \mathcal{P}^T\mathbf{x}_i^o$, $Q_i = \mathcal{P}^T\mathbf{q}_i$, $\mathcal{Y}_i = \mathcal{P}^T\mathbf{y}_i$, $\boldsymbol{\epsilon}_i = \mathcal{P}^T\mathbf{e}_i$, and hence

$$\mathcal{Y}_i = \mathcal{X}_i^o + \boldsymbol{\epsilon}_i; \quad \mathcal{X}_i^o = a\mathcal{X}_{i-1}^o + Q_i \tag{3.46}$$

$$\ddot{\mathcal{G}}_i = \underbrace{\mathcal{A}^T\mathcal{P}^T\mathcal{P}\left(diag\{(I_1 + \ddot{G}_{1,i})^{-1}, \ldots, (I_N + \ddot{G}_{N,i})^{-1}\}\right)}_{\text{Network DLS-LGT Filters.}}. \tag{3.47}$$

Let $r = r_k$ for all $k$. Rewrite Eq.(3.45) for the whole network as

$$\begin{aligned}\mathcal{X}_i &= \frac{1}{1+r}\ddot{\mathcal{G}}_i\mathcal{Y}_i + \frac{r}{1+r}\mathcal{X}_{i-1} \\ &= \frac{1}{1+r}\ddot{\mathcal{G}}_i\mathcal{X}_i^o + \frac{1}{1+r}\ddot{\mathcal{G}}_i\boldsymbol{\epsilon}_i + \frac{r}{1+r}\mathcal{X}_{i-1}.\end{aligned} \tag{3.48}$$

Let $\mathbb{E}\mathcal{G}_i = \mathcal{G}$. Then, we obtain the expected value of (3.48)

$$\mathbb{E}\mathcal{X}_i = \frac{r}{1+r}\mathbb{E}\mathcal{X}_{i-1} + \frac{1}{1+r}\ddot{\mathcal{G}}\mathbb{E}\mathcal{X}_i^o. \tag{3.49}$$

For $\frac{r}{1+r} < 1$, Eq.(3.49) can be written in steady state as,

$$\begin{aligned}\mathbb{E}\mathcal{X}_\infty &= \lim_{i\to\infty}\mathbb{E}\mathcal{X}_i \\ &= \bar{\mathcal{X}}^o - \underbrace{\left(\mathcal{I} - \ddot{\mathcal{G}}\right)\bar{\mathcal{X}}^o}_{\substack{\text{True signal rejected} \\ \text{by DLS-LGT filters.}}}.\end{aligned} \tag{3.50}$$

where $\bar{\mathcal{X}}^o = \mathcal{P}^T \bar{\mathbf{x}}^o$ and $\mathcal{I}$ is an $\sum_{k=1}^{N} N_k \times \sum_{k=1}^{N} N_k$ identity matrix. Again, notice that the estimator converges to the true signal minus any component of the true signal rejected by the DLS-LGT filters in the steady state. Therefore, the proposed DLS-LGT estimators is bias. In the next sections, we will study the bias and its mean-squared behavior.

### 3.5.2 Error Recursion and Bias

Introduce the network's learning error at time $i$, $\tilde{\mathcal{X}}_i$ defined as,

$$\begin{aligned}
\tilde{\mathcal{X}}_i &= \mathcal{X}_i^o - \mathcal{X}_i \\
&= \frac{r}{1+r}\tilde{\mathcal{X}}_{i-1} - \frac{r}{1+r}\mathcal{X}_{i-1}^o + \mathcal{X}_i^o - \frac{1}{1+r}\ddot{\mathcal{G}}_i\mathcal{X}_i^o - \frac{1}{1+r}\ddot{\mathcal{G}}_i\boldsymbol{\epsilon}_i \\
&= \frac{r}{1+r}\tilde{\mathcal{X}}_{i-1} + \left(\mathcal{I} - \frac{1}{1+r}\ddot{\mathcal{G}}_i - \frac{r}{a(1+r)}\mathcal{I}\right)\mathcal{X}_i^o + \frac{r}{a(1+r)}Q_i - \frac{1}{1+r}\ddot{\mathcal{G}}_i\boldsymbol{\epsilon}_i \\
&= \frac{r}{1+r}\tilde{\mathcal{X}}_{i-1} + \left(\frac{r-a(1+r)}{a(1+r)}\mathcal{I} - \frac{1}{1+r}\ddot{\mathcal{G}}_i\right)\mathcal{X}_i^o + \frac{r}{a(1+r)}Q_i - \frac{1}{1+r}\ddot{\mathcal{G}}_i\boldsymbol{\epsilon}_i. \qquad (3.51)
\end{aligned}$$

Let $\bar{Q} = \mathcal{P}^T\bar{\mathbf{q}}$ and $\mathbb{E}\boldsymbol{\epsilon}_i = 0$, the expected value of the above learning error becomes,

$$\mathbb{E}\tilde{\mathcal{X}}_i = \frac{r}{1+r}\mathbb{E}\tilde{\mathcal{X}}_{i-1} + \left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{1}{1+r}\ddot{\mathcal{G}}\right)\mathbb{E}\mathcal{X}_i^o + \frac{r}{a(1+r)}\mathbb{E}Q_i \qquad (3.52)$$

From (3.43), we have $\mathbb{E}Q_i = (1-a)\bar{\mathcal{X}}^o$. Hence, we conclude the estimator's bias in the steady state as

$$\begin{aligned}
\mathbb{E}\tilde{\mathcal{X}}_\infty &= \lim_{i\to\infty}\mathbb{E}\tilde{\mathcal{X}}_i \\
&= \underbrace{\left(\mathcal{I} - \ddot{\mathcal{G}}\right)\bar{\mathcal{X}}^o}_{\substack{\text{True signal rejected} \\ \text{by DLS-LGT filters.}}} < \infty. \qquad (3.53)
\end{aligned}$$

### 3.5.3 Mean-Square Stability

In this section, we will examine the variation of the learning error and its steady state behaviour. Using the learning error recursion from (3.51), we get the expected square of the network learning error $\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2$ as follows;

$$\begin{aligned}\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 = \frac{r^2}{(1+r)^2}\mathbb{E}\|\tilde{\mathcal{X}}_{i-1}\|^2 + tr\Bigg(&\frac{1}{(1+r)^2}\ddot{\mathcal{G}}\left(\mathbb{E}\epsilon_i\epsilon_i^T\right)\ddot{\mathcal{G}}^T + \frac{r^2}{a^2(1+r)^2}\mathbb{E}\left(Q_iQ_i^T\right)\\ &+\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)\mathbb{E}\left(\mathcal{X}_i^o\mathcal{X}_i^{o^T}\right)\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)^T\\ &+\frac{2r}{a(1+r)}\mathbb{E}\left(Q_i\mathcal{X}_i^{o^T}\right)\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)^T\\ &+\frac{2r}{1+r}\mathbb{E}\tilde{\mathcal{X}}_{i-1}\left(\mathbb{E}\mathcal{X}_i^{o^T}\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)+\frac{r}{a(1+r)}\mathbb{E}Q_i^T\right)\Bigg)\end{aligned} \tag{3.54}$$

We assume $\mathcal{X}_{i-1}^o$ and $Q_i$ are uncorrelated and $\mathbb{E}\left(Q_i\mathcal{X}_i^{o^T}\right) = \mathbb{E}\left(Q_iQ_i^T\right)$. For the ease of the analysis,we assume

$$\left(\mathbb{E}\mathcal{X}_i^o\right)\left(\mathbb{E}Q_i\right)^T = \left(\mathbb{E}Q_i\right)\left(\mathbb{E}Q_i^T\right);$$

$$\mathbb{E}\left(\left(\mathcal{X}_i^o-\mathbb{E}\mathcal{X}_i^o\right)\left(\mathcal{X}_i^o-\mathbb{E}\mathcal{X}_i^o\right)^T\right) = \mathbb{E}\left(\left(Q_i-\mathbb{E}Q_i\right)\left(Q_i-\mathbb{E}Q_i\right)^T\right)$$

From (3.52), we have the squared of the expected learning error as

$$\begin{aligned}\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)^T = \frac{r^2}{(1+r)^2}&\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)^T + \frac{r^2}{(1+r)^2}\left(\mathbb{E}Q_i\right)\left(\mathbb{E}Q_i\right)^T\\ &+\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)\left(\mathbb{E}\mathcal{X}_i^o\right)\left(\mathbb{E}\mathcal{X}_i^o\right)^T\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)^T\\ &+\frac{2r}{1+r}\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)\left(\mathbb{E}\mathcal{X}_i^o\right)\left(\mathbb{E}Q_i\right)^T\\ &+\frac{2r}{1+r}\mathbb{E}\tilde{\mathcal{X}}_i\left(\frac{r}{a(1+r)}\mathbb{E}Q_i^T+\mathbb{E}\mathcal{X}_i^{o^T}\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I}-\frac{1}{1+r}\ddot{\mathcal{G}}\right)^T\right).\end{aligned} \tag{3.55}$$

Comparing the expressions in (3.55) and (3.54), we can rewrite (3.54) as

$$\begin{aligned}\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 = \frac{r^2}{(1+r)^2}\mathbb{E}\|\tilde{\mathcal{X}}_{i-1}\|^2 + tr\Bigg(&\frac{1}{(1+r)^2}\ddot{\mathcal{G}}\mathcal{P}^T\Sigma\mathcal{P}\ddot{\mathcal{G}}^T \\ &+ \frac{r^2}{a^2(1+r)^2}\mathbb{E}\left(\left(Q_i - \mathbb{E}Q_i\right)\left(Q_i - \mathbb{E}Q_i\right)^T\right) \\ &+ \left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{1}{1+r}\ddot{\mathcal{G}}\right)\mathbb{E}\left(\left(Q_i - \mathbb{E}Q_i\right)\left(Q_i - \mathbb{E}Q_i\right)^T\right) \\ &\cdot\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{1}{1+r}\ddot{\mathcal{G}}\right)^T \\ &+ \frac{2r}{1+r}\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{1}{1+r}\ddot{\mathcal{G}}\right)\mathbb{E}\left(\left(Q_i - \mathbb{E}Q_i\right)\left(Q_i - \mathbb{E}Q_i\right)^T\right) \\ &+ \left(\mathbb{E}\tilde{\mathcal{X}}_i\right)\left(\mathbb{E}\tilde{\mathcal{X}}_i\right)^T - \frac{r^2}{(1+r)^2}\left(\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right)^T\Bigg)\end{aligned} \tag{3.56}$$

Hence, from (3.56), we write the steady state mean squared error as

$$\begin{aligned}\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 &= \lim_{i\to\infty}\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 \\ &\approx \underbrace{\|\mathbb{E}\tilde{\mathcal{X}}_\infty\|^2}_{\text{bias squared.}} + \underbrace{\frac{1}{1+2r}tr\left(\ddot{\mathcal{G}}\mathcal{P}^T\Sigma\mathcal{P}\ddot{\mathcal{G}}^T\right)}_{\substack{\text{errors due to the} \\ \text{measurement noise.}}} \\ &\quad \underbrace{+ tr\left(\left((1+r)\mathcal{I} - \ddot{\mathcal{G}}\right)\mathcal{P}^T L^\dagger \mathcal{P}\left((1+r)\mathcal{I} - \ddot{\mathcal{G}}\right)^T\right)}_{\text{errors due to non-stationary signal model.}}\end{aligned} \tag{3.57}$$

Let $MSD^r$ be the mean square deviation of the network with DLS-LGT algorithm. Then from (3.57), we conclude that,

$$MSD^r = \frac{1}{\sum_{k=1}^N N_k}\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 < \infty. \tag{3.58}$$

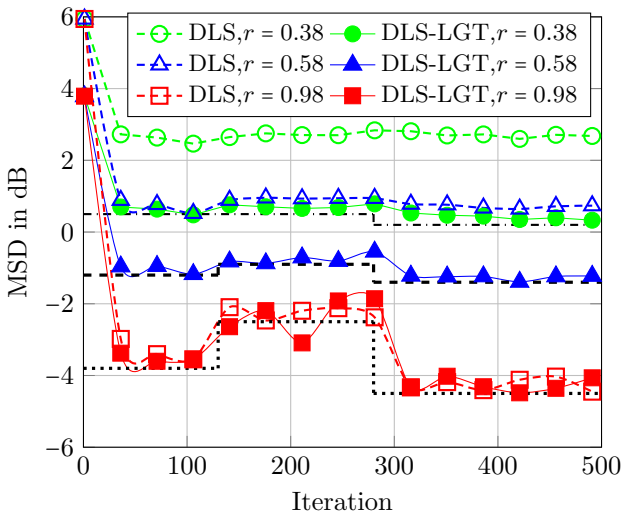
## 3.6 Experiments

### 3.6.1 Experiment on Synthetic Data

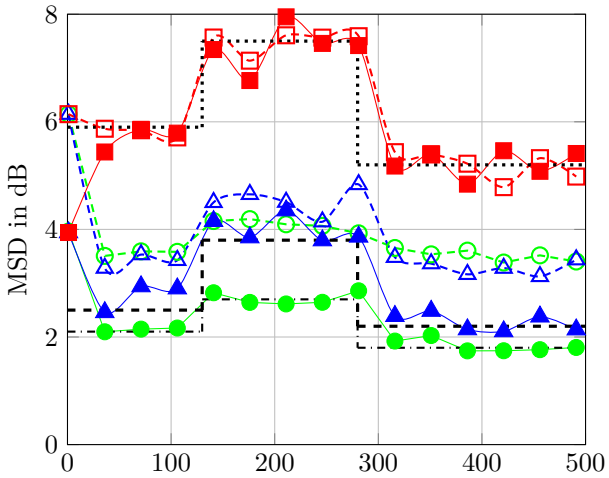
Consider a network with 100 nodes randomly distributed over a 2D plane. The network undirected Laplacian $L$ is formed based on the Euclidean distance between the nodes with the maximum number of neighbors of 6. We set $a_{\ell,k} = \frac{1}{N_k}$. The true graph signal is generated using (3.43) and $\mathbf{x}_o$ and $\mathbf{q}_i$ is drawn from a Gaussian distribution with mean of 10 and the variance of $0.001L^{\dagger}$. We tested the proposed LS-LGT algorithms for both slowing and rapidly varying signal for $a = 0.2$ and $a = 0.8$ in (3.43), respectively. We evaluated the proposed algorithm for time-varying network with both slowly and rapidly varying graph signal. We generated the graph signal using Eq.(3.43) with $a = 0.5$ and $\mathbf{q}_i$ is drawn from $0.01L^{\dagger}$ abd $L^{\dagger}$ for slowly and rapidly varying signal, respectively. We changed 10% of the edges at iteration, $i = 130$ and 280, respectively. We compared the MSD between the transitional dist.-RLS (DLS) [5] and the proposed DLS-LGT solution. Figure 3.6 (a) and (b) have the MSD comparisons between the proposed distributed least square strategy with local graph transform (DLS-LGT) and DLS for different $r$ values.

### 3.6.2 Experiment on Real-World Data

Similar to the experiment on Florida weather data in Section (3.3.2), we tested the LS-LGT algorithm on tracking the daily average temperature value from Section (3.3.2). We corrupted the temperature value with a zero-mean Gaussian noise for $SNR = -2dB$. We run the test for different $r$ values and the results are average over




(a)




(b)

Figure 3.6: MSD Comparison of time varying graph with DLS-LGT Alg. (5) and the traditional dist. LS for different value of $r$. The dashdotted, dashed and dotted lines are the theoritical MSD value of DLS-LGT Alg. for $r$ = 0.38, 0.58 and 0.98, respectively. The time varying true signal is generated using (3.43) with $a$ = 0.5, $\bar{\mathbf{q}}$ = 10 and the covariance of $\mathbf{q}_i$ are (a) $0.01L^{\dagger}$ and (b) $L^{\dagger}$, respectively. The graph changes 10% of the total edges at iteration, $i$ = 130 and 280.

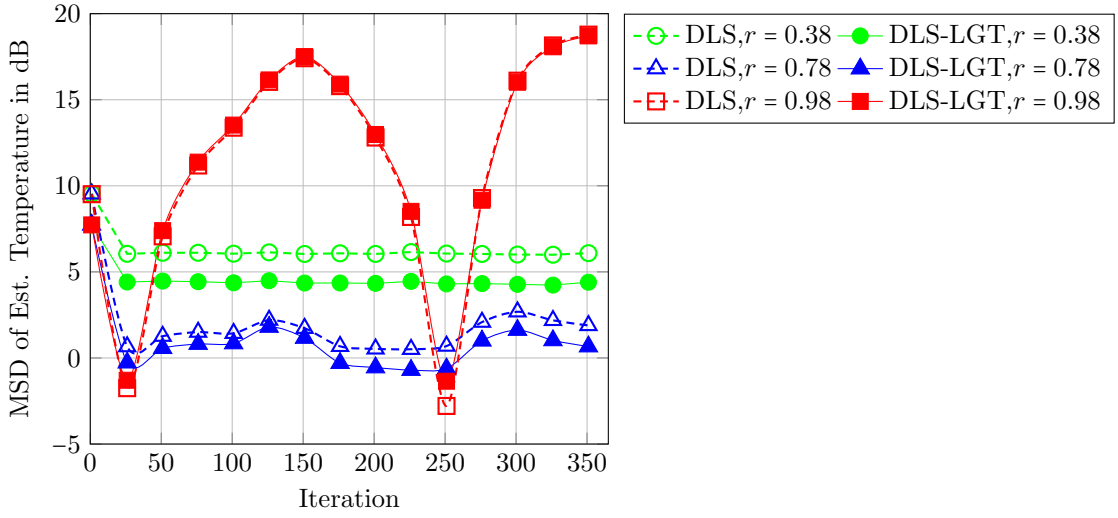



Figure 3.7: MSD Comparison of DLS-LGT Alg. (5) and traditional Dist. LS for different value of $r$ on Weather Data. The true temperature values are shown in 3.4 (b). We corrupted the true temperature with a zero-mean Gaussian noise. $SNR = -2dB$.

100 simulation. Figure 3.7 shows the MSD comparison for different value of $r$. The tracking performance of the LS-LGT Alg.5 is shown in Figure 3.8.

## 3.7 Conclusion

We examine the distributed estimation in LR-LS and LR-LMS for non-stationary signal in the time varying environment. We introduce the LGT notion to each node $k$ and design the LGT filter response without the global/centralized knowledge resulting the performance gain over the centralized solution. Next, we will study the LGT filter in estimation and tracking of graph signal in the present of outlier signal.

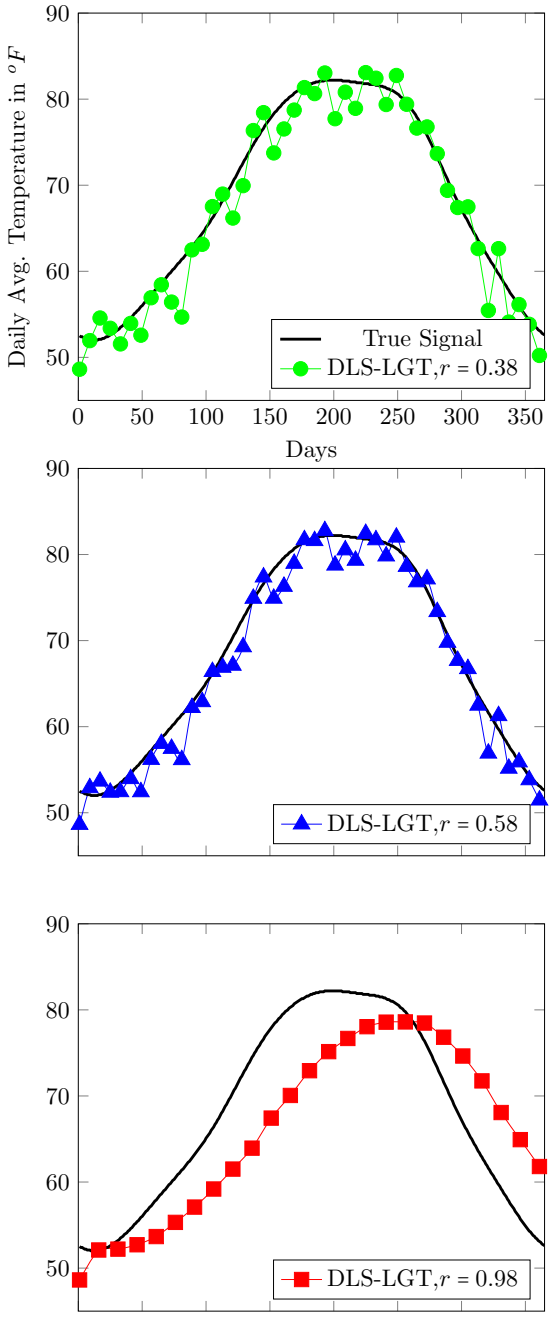



Figure 3.8: Tracking performance comparison of the average temperature recorded between Jan 1st, 2010 and Dec 31st, 2010 for different value of $r$. The Y-axis represents the average temperature value and the X-axis represents the days. The true temperature values are corrupted with a zero-mean Gaussian noise. $SNR = -2dB$.

# CHAPTER 4

# Signal and Outliers Estimation in Local Graph

## 4.1 Graph Signal and Outliers Estimation

In this chapter, we examine the fundamental challenge of estimating and tracking both non-stationary signals and outliers from noisy measurement in time-varying graphs. We assume that the non-stationary ground truth signal follows a random walk model in which the correlated random step is connected to the time-varying graph structure. We further assume that a random subset of nodes is perturbed by outlier signals that must be estimated as well. To address this challenge, we use the Local Graph Transform (LGT) and associated LGT-filters and incorporate it into an optimization-based adaptation and learning method.

Consider a sensor network with nodes $N$ and at time $i$, each node $k$ has the measurements $\{y_k(i), u_k(i)\}$, $k = 1, \ldots, N$. We assume that the measurement process at time $i$ is given by

$$y_k(i) = u_k(i)x_k^o(i) + o_k^o(i) + e_k(i). \tag{4.1}$$

where $x_k^o(i)$ is the node $k$'s ground truth signal at time $i$ to be estimated. Let $o_k^o(i)$ be a possibly present outlier/anomaly signal in node $k$'s measurement, and $e_k(i)$ is node $k$'s measurement noise. Suppose we consider the streams simultaneously by defining stacked vectors $\{\mathbf{y}_i, U_i\}$ in which $\mathbf{y}_i = \{y_1(i); \ldots; y_N(i)\}_{N\times 1}$, $U_i = diag\{u_1(i); \ldots; u_N(i)\}_{N\times N}$. Then network aggregate measurment is

$$\mathbf{y}_i = U_i \mathbf{x}_i^o + \mathbf{o}_i^o + \mathbf{e}_i, \tag{4.2}$$

where $\mathbf{e}_i = \{e_1(i); \ldots; e_N(i)\}_{N\times 1}$, $\mathbf{x}_i^o = \{x_1^o(i); \ldots; x_N^o(i)\}_{N\times 1}$ and $\mathbf{o}_i^o = \{o_1^o(i); \ldots; o_N^o(i)\}$. We assume that at time $i$ that the normalized graph Laplacian matrix $L_i$ describes the similarities among the ground truth signal at different nodes and the ground truth is a non-stationary signal that evolves according to a random walk model from (2.4). We consider $\mathbf{o}_i^o$ to be a sparse vector in that at any point in time, only a subset of random nodes have outlier contamination signals. The goal of the network is to recover and track both the true network signal $\mathbf{x}_i^o$ and the sparse outliers $\mathbf{o}_i^o$, with the recovered outlier signal of interst for subsequent processing, e.g., detection generalized likelihood ratio tests. So, the centralized cost to be minimized over the estimates $\mathbf{x}_i$ and $\mathbf{o}_i$ is

$$\begin{aligned} &J^g(\mathbf{x}_i, \mathbf{o}_i) = \mathbb{E}\|\mathbf{y}_i - U_i\mathbf{x}_i - \mathbf{o}_i\|^2 + \beta h(\mathbf{x}_i) + \alpha\|\mathbf{o}_i\|_1 \\ &\text{subject to } \mathbb{E}\mathbf{y}_i = \mathbb{E}[U_i\mathbf{x}_i + \mathbf{o}_i], \end{aligned} \tag{4.3}$$

where $\beta$ and $\alpha$ are non-negative parameter and $\|.\|_1$ is the $\ell_1$ norm. The convex cost $h(\mathbf{x}_i)$ in (4.3) penalizes the total variation across the network, i.e.,

$$h(\mathbf{x}_i) = \mathbf{x}_i^T L_i \mathbf{x}_i. \tag{4.4}$$

## 4.2 Signal and Outliers Estimation in Local Graph with LMS Strategy

We are interested in the distributed solution for the global cost in (4.3) where each node $k$ estimates its true signal $x_k^o(i)$ and the possible outlier/anomaly signal $o_k^o(i)$ with the aid of message passing with its set of neighbors $\mathcal{N}_k(i)$ at time $i$. Note that node $k$ is itself in this set and let $N_k(i) = |\mathcal{N}_k(i)|$. For readability, we will derive the distributed algorithm assuming a fixed graph Laplacian, and suppress the time index $i$ in the graph/network terms in the following discussion.

Before proceeding, note that if the global estimate $\mathbf{x}_i$ is needed, then we assume that a separate collating and forwarding protocol is used. Note that each node in the network has a unique global ID and we use $k$ and $\ell$ to refer to the global IDs for nodes in the network. We assume each node $k$ does not have information about its neighbors' global IDs and so node $k$ has assigned its neighbors its own local numbering system, i.e., as $k_1, \ldots, k_{N_k-1}$ where $N_k$ is the total number of node $k$'s neighbors (note that node $k$ is listed first in the set $\mathcal{N}_k$). Let $\mathbf{x}_{k,i} = \{x_k(i); x_{k_1}(i); \ldots; x_{k_{N_k-1}(i)}\}_{N_k \times 1}$ be the $N_k \times 1$ vector of node $k$'s estimates of its own signal and the signals of its neighbors. We define each node $k$'s local penalty function as

$$h_k(\mathbf{x}_{k,i}) = \mathbf{x}_{k,i}^T S_k \mathbf{x}_{k,i} \tag{4.5}$$

where $S_k$ is an $N_k \times N_k$ local Laplacian network of node $k$, i.e., the graph Laplacian solely from the vantage point of node $k$, meaning a star network. This local penalty

measures the similarity of the signal estimates in $\mathcal{N}_k$. One can verify that

$$\begin{aligned} h(\mathbf{x}_i^o) &= \sum_{k=1}^{N} h_k\left(\mathbf{x}_{k,i}^o\right) \\ &= \sum_{k=1}^{N} \mathbf{x}_{k,i}^{o^T} S_k \mathbf{x}_{k,i}^o \\ &= \mathbf{x}_i^{o^T} \left(\sum_{k=1}^{N} P_k S_k P_k^T\right) \mathbf{x}_i^o \end{aligned} \tag{4.6}$$

where $P_k$ is an $N \times N_k$ corresponding permutation matrix. We write the global cost in (4.3) as a sum of local costs as

$$J^g(\mathbf{x}_i, \mathbf{o}_i) = \sum_{k=1}^{N} J_k(x_k(i), o_k(i)) \tag{4.7}$$

where

$$\begin{aligned} J_k(x_k(i), o_k(i)) &= \mathbb{E}|y_k(i) - u_k(i)x_k(i) - o_k(i)|^2 + \sum_{l \in \mathcal{N}_k} a_{\ell,k}\beta_\ell h_\ell(\mathbf{x}_{\ell,i}) + \alpha_k|o_k(i)| \\ \text{subject to } \mathbb{E}y_k(i) &= \mathbb{E}[u_k(i)x_k(i) + o_k(i)], \end{aligned} \tag{4.8}$$

where $\beta_\ell$, $\alpha_k$ and $b_{\ell,k}$ are non-negative coefficients and $b_{\ell,k} = 0$ for $\ell \notin \mathcal{N}_k$ and $\sum_{\ell \in \mathcal{N}_\ell} b_{\ell,k} = 1$ and $x_k(i)$ and $o_k(i)$ are the estimates of $x_k^o(i)$ and $o_k^o(i)$ respectively. Note that for $\ell \in \mathcal{N}_k$, the vector $\mathbf{x}_{\ell,i}$ contains $x_k(i)$. Using the auxiliary variable $z_k$, we write the constrained cost in (4.8) as

$$\begin{aligned} \mathcal{J}_k(x_k(i), o_k(i), z_k) &= \mathbb{E}|y_k(i) - u_k(i)x_k(i) - o_k(i)|^2 + \sum_{\ell \in \mathcal{N}_k} a_{\ell,k}\beta_\ell h_\ell(\mathbf{x}_{\ell,i}) + \alpha_k|o_k(i)| \\ &\quad + \gamma_k z_k(\mathbb{E}y_i - \mathbb{E}u_k(i)x_k(i) - \mathbb{E}o_k(i)). \end{aligned} \tag{4.9}$$

where $\gamma_k$ is non negative coefficient. Each node $k$ solves (4.9) as [63, 64],

$$x_k(i) = \underset{x_k(i)}{\operatorname{argmin}} \mathcal{J}_k\Big(x_k(i), o_k(i-1), z_k(i-1)\Big) \tag{4.10}$$

$$o_k(i) = \underset{o_k(i)}{\operatorname{argmin}} \mathcal{J}_k\Big(x_k(i), o_k(i), z_k(i-1)\Big) \tag{4.11}$$

$$z_k(i) = \underset{z_k}{\operatorname{argmin}} \mathcal{J}_k(x_k(i), o_k(i), z_k\Big). \tag{4.12}$$

Guided by [4], we solve (4.10) as

$$\begin{aligned} x_k(i) = x_k(i-1) + \mu_k u_k^*(i) \left(y_k(i) - u_k(i)x_k(i-1) - o_k(i-1) + \gamma_k z_k(i-1)\right) \\ + \mu_k \sum_{l \in \mathcal{N}_k} a_{\ell,k}\beta_\ell \nabla_{x_k(i)} h_\ell\left(\mathbf{x}_{\ell,i-1}\right). \end{aligned} \tag{4.13}$$

where $^*$ denotes the complex conjugate. Note that

$$\beta_\ell \nabla_{x_k(i)} h_\ell(\mathbf{x}_{\ell,i}) = \mathbf{p}_{\ell,k}^T\Big(\beta_\ell L_\ell \mathbf{x}_{\ell,i-1}\Big). \tag{4.14}$$

Without loss of generality, say that node $\ell$ assigns node $k$ as its second neighbor, then $\mathbf{p}_{\ell,k}$ is an $N_\ell \times 1$ vector with all zeros and one at the second element. Let $\psi_k(i)$ be the intermediate estimate of node $k$. Then

$$\phi_k(i) = x_k(i-1) + \mu_k u_k^*(i) \left(y_k(i) - u_k(i)x_k(i-1) - o_k(i-1) + \gamma_k z_k(i-1)\right). \tag{4.15}$$

Since $\phi_k(i)$ contains more information about $x_k^o(i)$ than $x_k(i-1)$, with $\boldsymbol{\phi}_{\ell,i} = \{\phi_{\ell_1}(i); \ldots; \phi_{\ell_{N_\ell}}(i)\}_{N_\ell \times 1}$ and from (3.10),

$$\begin{aligned} x_k(i) &= \phi_k(i) - \mu_k \sum_{\ell \in \mathcal{N}_k} a_{\ell,k}\mathbf{p}_{\ell,k}^T\left(\beta_\ell L_\ell \boldsymbol{\phi}_{\ell,i}\right) \\ &= \mu_k \sum_{\ell \in \mathcal{N}_k} a_{\ell,k}\mathbf{p}_{\ell,k}^T\left(I_{N_\ell} - V_{k,i}\breve{\Theta}_{k,i}V_{k,i}^T\right)\boldsymbol{\phi}_{\ell,i}. \end{aligned} \tag{4.16}$$

where $\breve{\Theta}_{k,i} = diag\{\breve{\theta}_{k,1}(i), \ldots, \breve{\theta}_{k,N_k}(i)\}$ and $\theta_{k,n}(i)$ is denoted as an LGT filter response and From Section 3.1.2, here we design the LGT filter response $w_{k,n}(i)$ as

$$\breve{\theta}_{k,n}(i) = \max\left\{1 - \frac{\zeta_{k,n}}{\boldsymbol{\Phi}_{k,i}(\lambda_{k,n})^2}, 0\right\} \leq \breve{\theta}_{max} \tag{4.17}$$

where $\boldsymbol{\Phi}_{k,i} = V_k^T \boldsymbol{\phi}_{k,i}$. Let $\breve{G}_{k,i} = V_{k,i}\breve{\Theta}_{k,i}V_{k,i}^T$. Note that $\breve{G}_{k,i}$ is an $N_k \times N_k$ is each node $k$'s local graph operator. We express (4.16) in two steps as

$$\boldsymbol{\varphi}_{k,i} = \underbrace{\left(I_k - \breve{G}_{k,i}\right)}_{\text{DLMS-LGT filter.}} \boldsymbol{\phi}_{k,i} \tag{4.18}$$

$$x_k(i) = \sum_{\ell \in \mathcal{N}_k} a_{\ell,k} \mathbf{p}_{\ell,k}^T \boldsymbol{\varphi}_{\ell,i} \tag{4.19}$$

where $a_{\ell,k} = \mu_k b_{\ell,k}$ and $\boldsymbol{\phi}_{\ell,i} = \{\phi_{\ell_1}(i), \ldots, \phi_{\ell_{N_\ell}}(i)\}_{N_\ell \times 1}$ and $I_k$ is an $N_k \times N_k$ identity matrix. If $\breve{G}_k = \beta_k S_k$ then the method reverts back to the standard local penalty. We now solve (4.11) as in [56], i.e.,

$$o_k(i) = o_k(i-1) + \mu'_k \left(y_k(i) - u_k(i)x_k(i) - o_k(i-1) + \gamma_k z_k(i-1)\right) - \alpha'_k sgn\{o_k(i-1)\} \tag{4.20}$$

where $\mu'_k$ is a positive step size and $\alpha'_k = \mu'_k \alpha_k$ and $sgn\{.\}$ returns the sign of the argument variable. One can interpret the outlier estimator as processing the components rejected by each node $k$'s DLMS-LGT filter, i.e., the residual mix of noise and possible outliers. Then, we update the auxiliary variable $z_k(i)$ from (4.12) as

$$z_k(i) = z_{k,i-1} + \gamma_k(y_k(i) - u_k(i)x_k(i) - o_k(i)), \tag{4.21}$$

where $\gamma_k$ is a non-negative coefficient. Algorithm 6 gives the complete algorithm for distributed graph signal and outlier estimation.

## 4.3 Performance of DLMS-LGT outlier

### 4.3.1 Stability and Mean Convergence

Our objective is to study the steady state behavior of the proposed LGT method in graph signal and outlier estimation. For the sake of the simplicity of the analysis, we

**Algorithm 6** DLMS-LGT and Outliers Estimation

Initialize $x_k(0) = y_k(0), o_k(0) = 0$ and $z_k(0) = 0, \forall k$ and node $k$ has $\zeta_{k,n}, n \in \{1, \dots N_k\}$.
**For** $i > 0$ and $k = 1$ to $N$ **do**

1. Each node $k$ has $\{y_k(i), u_k(i)\}$.
2. $\phi_k(i) = x_k(i-1) + \mu_k u_k^*(i) \left(y_k(i) - u_k(i)x_k(i-1) - o_k(i-1) + \gamma_k z_k(i-1)\right).$
3. Exchange $\{\phi_k(i)\}$ with the neighbors.
4. $\boldsymbol{\phi}_{k,i} = \{\phi_{k_1}(i); \dots; \phi_{k_{N_k}}(i)\}$
5. Find the LGT filter responses.

   **For** $n = 1$ to $N_k$ **do**

   $\breve{\theta}_{k,n}(i) = \max\left\{1 - \frac{1\zeta_n}{\boldsymbol{\Phi}_{k,i}(\alpha_n)^2}, 0\right\} \leq \breve{\theta}_{max}$, where $\boldsymbol{\Phi}_{k,i} = V_k^T \boldsymbol{\phi}_{k,i}$

   **End.**

   $\breve{\Theta}_{k,i} = diag\{\breve{\theta}_{k,1}(i), \dots, \breve{\theta}_{k,N_{k,i}}(i)\}.$
6. $\boldsymbol{\varphi}_{k,i} = (I_{N_k} - \breve{G}_{k,i})\boldsymbol{\phi}_{k,i}$ where $\breve{G}_{k,i} = V_k \breve{\Theta}_{k,i} V_k^T$.
7. Exchange $\varphi_{\ell,k}(i) = \mathbf{p}_{\ell,k}^T \boldsymbol{\varphi}_{k,i}$ with the neighbors.
8. $x_k(i) = \sum_{\ell \in \mathcal{N}_k} a_{\ell,k} \mathbf{p}_{\ell,k}^T \boldsymbol{\varphi}_{\ell,i}$.
9. $o_k(i) = o_k(i-1) + \mu_k' \left(y_k(i) - u_k(i)x_k(i) - o_k(i-1) + \gamma_k z_k(i-1)\right) - \alpha_k' sgn\{o_k(i-1)\}$.
10. $z_k(i) = z_{k,i-1} + \gamma_k(y_k(i) - u_k(i)x_k(i) - o_k(i))$.

**End.**

assume the stationary outliers, i.e, $\mathbf{o}_i^o = \mathbf{o}^o$ for all $i$. First, we examine the convergence of the LGT estimators. Let $\mathbf{x}_i = \{x_1(i); \ldots; x_N(i)\}$

$\mathbf{o}_i = \{o_1(i); \ldots; o_N(i)\}$

$R_{u,i} = diag\{u_1^*(i)u_1(i); \ldots; u_N^*(i)u_N(i)\}$,

$\mu = \mu_k = \mu_k'$, $\gamma = \gamma_k$ and $\alpha' = \alpha_k$ for all $k$, $\mathbf{z}_i = \{z_1(i); \ldots; z_n(i)\}$. Let $A = \{a_{\ell,k}\}$ be the weighted adjacency matrix where $[A]_{\ell,k} = a_{\ell,k}$, $\sum_{\ell=1}^{N} a_{\ell,k} = 1$, $a_{\ell,k} = 0, \forall \ell \notin N_k$. Let $\mathcal{P}$ be the $N \times \prod_{k=1}^{N} N_k$ suitable permutation matrix and

$$\breve{G}_i = \underbrace{A^T \left(I_N - \mathcal{P}.diag\{\breve{G}_{1,i}, \ldots, \breve{G}_{N,i}\}.\mathcal{P}^T\right)}_{\text{Network DLMS-LGT filters.}}. \tag{4.22}$$

From (4.15), (4.18) and (4.19), we write the DLMS- LGT method in a matrix recursion form as,

$$\mathbf{x}_i = \breve{G}_i \left(I_N - \mu R_{u,i}\right) \mathbf{x}_{i-1} + \mu \breve{G}_i \left(R_{u,i}\mathbf{x}_i^o + U_i^*\mathbf{o}^o + U_i^*\mathbf{e}_i - U_i^*\mathbf{o}_{i-1} + \gamma U_i^*\mathbf{z}_{i-1}\right). \tag{4.23}$$

Notice from (4.18), (4.19), (4.20) and (4.21) that $\mathbf{o}_{i-1}$ and $\mathbf{z}_{i-1}$ are the weighted sum of the rejected signal from the LGT filters, therefore, we ignore $\breve{G}_i U_i^*(\mathbf{o}_{i-1} + \gamma \mathbf{z}_{i-1})$ in (4.23).

$$\mathbf{x}_i \approx \breve{G}_i(I_N - \mu R_{u,i})\mathbf{x}_{i-1} + \mu \breve{G}_i\Big(R_{u,i}\mathbf{x}_i^o + U_i^*\mathbf{o}^o + U_i^*\mathbf{e}_i\Big). \tag{4.24}$$

Let $\mathbb{E}\breve{G}_i = \breve{G}$, $\mathbb{E}R_{u,i} = R_u$ and $\mathbb{E}U_i = U$. Taking expectation to the both side of the above equation we have,

$$\mathbb{E}\mathbf{x}_i = \breve{G} \left(I_N - \mu R_u\right) \mathbb{E}\mathbf{x}_{i-1} + \mu \breve{G} \left(R_u\mathbf{x}_i^o + U * \mathbf{o}^o\right). \tag{4.25}$$

To ensure the stability of the above recursion, we need the following condition of

$$\|\breve{G} \left(I_N - \mu R_u\right) \| < 1$$

$$\rho(\breve{G}) \left|1 - \mu\rho(R_u)\right| < 1 \tag{4.26}$$

where $rho(.)$ is the spectral radius of a matrix. From (4.18) and (4.22), we have $\rho(\breve{G}) < |1 - \breve{\theta}_{max}|$. From (4.26), we limit the LGT filter response as

$$0 \leq \breve{\theta}_{max} < \frac{1 + |1 - \mu\rho(R_u)|}{|1 - \mu\rho(R_u)|} \tag{4.27}$$

We also express each node $k$'s outlier estimation and auxiliary variable update equation in (4.20) and (4.21) in a matrix recursion form as

$$\mathbf{o}_i = (1-\mu)\mathbf{o}_{i-1} + \mu\left(U_i\mathbf{x}_i^o + \mathbf{o}^o + \mathbf{e}_i - U_i\mathbf{x}_i + \gamma\mathbf{z}_{i-1}\right) - \alpha' sgn\{\mathbf{o}_{i-1}\}; \tag{4.28}$$

$$\mathbf{z}_i = \mathbf{z}_{i-1} + \gamma\mu\left(U_i\mathbf{x}_i^o + \mathbf{o}^o + \mathbf{e}_i - U_i\mathbf{x}_i + \mathbf{o}_i\right). \tag{4.29}$$

For simplicity, we assume $\alpha' << 1$ and we omit $sgn\{\mathbf{o}_{i-1}\}$ term in (4.28) in later analysis.

$$\mathbb{E}\mathbf{o}_i = (1-\mu)\mathbb{E}\mathbf{o}_{i-1} + \mu\left(U\mathbb{E}\mathbf{x}_i^o + \mathbf{o}^o - U\mathbb{E}\mathbf{x}_i + \gamma\mathbb{E}\mathbf{z}_{i-1}\right) \tag{4.30}$$

$$\mathbb{E}\mathbf{z}_i = c\mathbb{E}\mathbf{z}_{i-1} + \gamma\mu\left(U_i\mathbb{E}\mathbf{x}_i^o + \mathbb{E}\mathbf{o}^o + \mathbb{E}\mathbf{e}_i - U\mathbb{E}\mathbf{x}_i - \mathbb{E}\mathbf{o}_i\right). \tag{4.31}$$

We limit $0 < \mu < 2$ in (4.30) and introduce a variable $|c| < 1$ in (4.31) for the stability of Eq. (4.30) and (4.31), respectively. We express Eq.(4.25), (4.30) and (4.31) as a block matrix recursion , i.e.,

$$\begin{bmatrix} \mathbb{E}\mathbf{x}_i \\ \mathbb{E}\mathbf{o}_i \\ \mathbb{E}\mathbf{z}_i \end{bmatrix} \approx \begin{bmatrix} \breve{G}(I_N - \mu R_u) & \emptyset & \emptyset \\ -\mu U & (1-\mu)I_N & \mu\gamma I_N \\ -\gamma U & -\gamma I_N & cI_N \end{bmatrix} \begin{bmatrix} \mathbb{E}\mathbf{x}_{i-1} \\ \mathbb{E}\mathbf{o}_{i-1} \\ \mathbb{E}\mathbf{z}_{i-1} \end{bmatrix} + \begin{bmatrix} \mu\breve{G}(R_u\mathbb{E}\mathbf{x}_i^o + U^*\mathbf{o}^o) \\ \mu(U\mathbb{E}\mathbf{x}_i^o + \mathbf{o}^o) \\ \gamma(U\mathbb{E}\mathbf{x}_i^o + \mathbf{o}^o) \end{bmatrix} \tag{4.32}$$

Let $\mathbb{E}\mathbf{x}_i^o = \bar{\mathbf{x}}^o$ Then, in steady state, Eq. (4.32) becomes

$$\begin{bmatrix} \mathbb{E}\mathbf{x}_\infty \\ \mathbb{E}\mathbf{o}_\infty \\ \mathbb{E}\mathbf{z}_\infty \end{bmatrix} = \lim_{i\to\infty} \begin{bmatrix} \mathbb{E}\mathbf{x}_i \\ \mathbb{E}\mathbf{o}_i \\ \mathbb{E}\mathbf{z}_i \end{bmatrix}$$

$$= \begin{bmatrix} I_N - \breve{G}(I_N - \mu R_u) & \emptyset & \emptyset \\ \mu U & \mu I_N & -\mu\gamma I_N \\ \gamma U & \gamma I_N & (1-c)I_N \end{bmatrix}^{-1} \begin{bmatrix} \mu\breve{G}(R_u\bar{\mathbf{x}}^o + U^* o^o) \\ \mu(U\bar{\mathbf{x}}^o + \mathbf{o}^o) \\ \gamma(U\bar{\mathbf{x}}^o + \mathbf{o}^o) \end{bmatrix} \tag{4.33}$$

From block matrix inversion equality, one can write the inverse of a block matrix as,

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]^{-1} = \left[\begin{array}{c|c} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ \hline -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{array}\right] \tag{4.34}$$

Using the block matrix inversion from above, we find the matrix inverse from (4.33).

Let $A = I_N - \breve{G}(I_N - \mu R_u)$, $B = \{\emptyset \quad \emptyset\}$, $C = \{\mu U; \gamma U\}$ and

$$D = \begin{bmatrix} \mu I_N & -\mu\gamma I_N \\ \gamma I_N & (1-c)I_N \end{bmatrix}; \quad D^{-1} = \begin{bmatrix} \frac{1-c}{\mu(1-c+\gamma^2)} I_N & \frac{\gamma}{1-c+\gamma^2} I_N \\ \frac{-\gamma}{\mu(1-c+\gamma^2)} I_N & \frac{1}{1-c+\gamma^2} I_N \end{bmatrix} \tag{4.35}$$

From the expression in (4.34), we have

$$\begin{bmatrix} I_N - \breve{G}(I_N - \mu R_u) & \emptyset & \emptyset \\ \mu U & \mu I_N & -\mu\gamma I_N \\ \gamma U & \gamma I_N & (1-c)I_N \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} \Big(I_N - \breve{G}(I_N - \mu R_u)\Big)^{-1} & \emptyset & \emptyset \\ -U\Big(I_N - \breve{G}(I_N - \mu R_u)\Big)^{-1} & \frac{1-c}{\mu(1-c+\gamma^2)} I_N & \frac{\gamma}{1-c+\gamma^2} I_N \\ \emptyset & \frac{-\gamma}{\mu(1-c+\gamma^2)} I_N & \frac{1}{1-c+\gamma^2} I_N \end{bmatrix} \tag{4.36}$$

Now, using (4.33) and (4.36), as $c \to 1$, we conclude that

$$\mathbb{E}\mathbf{x}_\infty = \bar{\mathbf{x}}^o - \underbrace{\left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o}_{\text{True graph signal rejected by DLMS-LGT filters}} + \underbrace{\mu\left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\breve{G}U^*\mathbf{o}^o}_{\text{Outlier accepted by DLMS-LGT filters}} \tag{4.37}$$

$$\mathbb{E}\mathbf{o}_\infty = o^o - \underbrace{\mu\left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\breve{G}U^*\mathbf{o}^o}_{\text{Outlier accepted by DLMS-LGT filters}} + \underbrace{U\left(I - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o}_{\text{True graph signal rejected by DLMS-LGT filters}}. \tag{4.38}$$

and $\mathbb{E}\mathbf{z}_\infty = 0$. Notice that $\mathbb{E}\mathbf{x}_\infty$ converges to the sum of the true graph signal and outliers filtered by the DLMS-LGT filters and $\mathbb{E}\mathbf{o}_\infty$ converges to the sum of the true outliers and the other two extra terms; the outliers filtered by the DLMS-LGT filters and the rejected component of the true graph signal from the DLMS-LGT filters. When the filters can differentiate between the energy of the frequency representation of the true graph signal and the outliers,i.e.,

$$\left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o \approx 0; \quad \left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\breve{G}U^*\mathbf{o}^o \approx 0. \tag{4.39}$$

then, we can conclude that

$$\mathbb{E}\mathbf{x}_\infty \to \bar{\mathbf{x}}^o; \quad \mathbb{E}\mathbf{o}_\infty \to \mathbf{o}^o. \tag{4.40}$$

For example, the energy of the frequency representation of the true graph signal and outliers lie in the lower and higher frequencies of the LGT, respectively and we use the low pass LGT filters, then, the proposed LGT-based distributed estimators become asymptotically unbiased. Figure 4.1 has the example of band-limited graph signal and outliers which resides in the different node $k$'s LGT frequencies. Then we can use the low pass LGT filter as in Figure 4.2 to achieve the unbiased estimators.

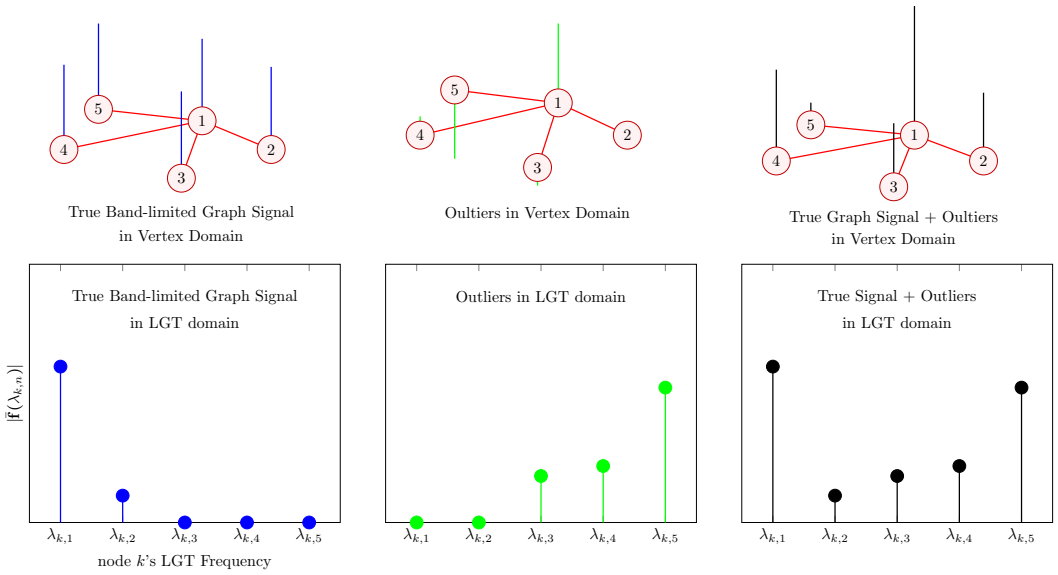



Figure 4.1: Band Limited Graph Signal and Outliers.

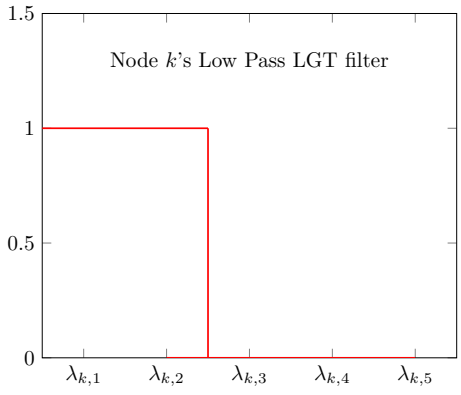



Figure 4.2: Node $k$'s LGT low-pass filter.

### 4.3.2 Error Recursion and Bias

Introduce $\tilde{x}_k(i)$ and $\tilde{o}_k(i)$ as the signal and outlier estimation errors of node $k$ at time $i$ respectively, i.e.,

$$\tilde{x}_k(i) = x_k^o(i) - x_k(i), \quad \tilde{o}_k(i) = o_k^o(i) - o_k(i). \tag{4.41}$$

Let $\tilde{\mathbf{x}}_i = \{\tilde{x}_1(i), \ldots, \tilde{x}_N(i)\}$,

$\tilde{\mathbf{o}}_i = \{\tilde{o}_1(i), \ldots, \tilde{o}_N(i)\}$,

Here, we explore the condition for the stability of the these network error vectors. With (4.24) and (4.28), the error vectors become

$$\begin{bmatrix} \tilde{\mathbf{x}}_i \\ \tilde{\mathbf{o}}_i \end{bmatrix} = \begin{bmatrix} \breve{G}_i \left(I_N - \mu R_{u,i}\right) & \emptyset \\ -\mu U_i & (1-\mu) I_N \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}}_{i-1} \\ \tilde{\mathbf{o}}_{i-1} \end{bmatrix} + \begin{bmatrix} \left(I_N - \breve{G}_i\right) \mathbf{x}_i^o - \mu \breve{G}_i U_i^* \mathbf{o}^o \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \breve{G}_i (I_N - \mu R_{u,i}) \mathbf{q}_i \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mu \breve{G}_i U_i^* \mathbf{e}_i \\ \mu \mathbf{e}_i \end{bmatrix} \tag{4.42}$$

where $\mathbf{0}$ is an $N \times 1$ all zero vector. We assume $\mathbb{E}\mathbf{e}_i$ and $\mathbb{E}\mathbf{q}_i = \mathbf{0}$. We can express the expected value of (4.42) as

$$\begin{bmatrix} \mathbb{E}\tilde{\mathbf{x}}_i \\ \mathbb{E}\tilde{\mathbf{o}}_i \end{bmatrix} = \begin{bmatrix} \breve{G} \left(I_N - \mu R_u\right) & \emptyset \\ -\mu U & (1-\mu) I_N \end{bmatrix} \begin{bmatrix} \mathbb{E}\tilde{\mathbf{x}}_{i-1} \\ \mathbb{E}\tilde{\mathbf{o}}_{i-1} \end{bmatrix} + \begin{bmatrix} \left(I_N - \breve{G}\right) \bar{\mathbf{x}}^o - \mu \breve{G} U^* \mathbf{o}^o \\ \mathbf{0} \end{bmatrix} \tag{4.43}$$

As $i \to \infty$, Eq.(4.43) becomes

$$\begin{aligned}
\begin{bmatrix} \mathbb{E}\tilde{\mathbf{x}}_\infty \\ \mathbb{E}\tilde{\mathbf{o}}_\infty \end{bmatrix} &= \lim_{i\to\infty} \begin{bmatrix} \mathbb{E}\tilde{\mathbf{x}}_i \\ \mathbb{E}\tilde{\mathbf{o}}_i \end{bmatrix} \\
&= \begin{bmatrix} I_N - \breve{G}\left(I_N - \mu R_u\right) & \emptyset \\ \mu U & \mu I_N \end{bmatrix}^{-1} \begin{bmatrix} \left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o - \mu\breve{G}U^*\mathbf{o}^o \\ \mathbf{0} \end{bmatrix} \\
&= \begin{bmatrix} \left(I_N - \breve{G}(I_N - \mu R_u)\right)^{-1} & \emptyset \\ -U\left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1} & \frac{1}{\mu} I_N \end{bmatrix} \begin{bmatrix} \left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o - \mu\breve{G}U^*\mathbf{o}^o \\ \mathbf{0} \end{bmatrix}
\end{aligned} \tag{4.44}$$

Notice that

$$\mathbb{E}\tilde{\mathbf{x}}_\infty = \underbrace{\left(I_N - \breve{G}(I_N - \mu R_u)\right)^{-1}\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o}_{\text{True graph signal rejected by DLMS-LGT filters.}} \underbrace{- \mu\left(I_N - \breve{G}\left(I_N - \mu R_u\right)\right)^{-1}\breve{G}U^*\mathbf{o}^o}_{\text{Outliers accepted by DLMS-LGT filters.}} < \infty \tag{4.45}$$

and $\mathbb{E}\tilde{\mathbf{o}}_\infty = -U\mathbb{E}\tilde{\mathbf{x}}_\infty$.

### 4.3.3 Mean-Square Stability

Now, let us evaluate the mean square stability of the error vectors in (4.42). Let $\breve{\mathcal{B}}_i = \breve{G}_i\left(I_N - \mu R_{u,i}\right)$. Using (4.42), we obtain the recursion for the mean square of the error vectors as follows;

$$\begin{aligned}
\mathbb{E}\|\tilde{\mathbf{x}}_i\|^2 = \mathbb{E}\|\breve{\mathcal{B}}_i\tilde{\mathbf{x}}_{i-1}\|^2 &+ tr\left(\mu^2\breve{G}U^*\mathbb{E}\left(\mathbf{e}_i\mathbf{e}_i^T\right)U\breve{G}^T + B\mathbb{E}\left(\mathbf{q}_i\mathbf{q}_i^T\right)B^T + \mu^2\breve{G}U^*\mathbf{o}^o\mathbf{o}^{oT}U\breve{G}^T\right) \\
&+ tr\left(\left(I_N - \breve{G}\right)\mathbb{E}\left(\mathbf{x}_i^o\mathbf{x}_i^{oT}\right)\left(I_N - \breve{G}\right)^T + 2\mu\breve{G}U^*\mathbf{o}^o\bar{\mathbf{x}}^{oT}\left(I_N - \breve{G}\right)^T\right) \\
&+ 2\ tr\left(\breve{\mathcal{B}}\mathbb{E}\tilde{\mathbf{x}}_{i-1}\left(\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o - \mu\breve{G}U^*\mathbf{o}^o\right)^T\right)
\end{aligned} \tag{4.46}$$

$$\mathbb{E}\|\tilde{\mathbf{o}}_i\|^2 = (1-\mu)^2\,\mathbb{E}\|\tilde{\mathbf{o}}_{i-1}\|^2 + \mu^2 tr\left(R_u\mathbb{E}\left(\tilde{\mathbf{x}}_{i-1}\tilde{\mathbf{x}}_{i-1}^T\right)\right) + \mu^2\Sigma - 2(1-\mu)tr\left(\left(\mathbb{E}\tilde{\mathbf{o}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}^T\right)\right). \tag{4.47}$$

Let $\mathbb{E}\breve{B}_i = \breve{B}$. From 4.43, we write the squared of the expected error $\mathbb{E}\tilde{x}_i$ as

$$\|\mathbb{E}\tilde{\mathbf{x}}_i\|^2 = \|\breve{\mathcal{B}}\mathbb{E}\tilde{\mathbf{x}}_{i-1}\|^2 + \|\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o\|^2 + \mu^2\|\breve{G}U^*\mathbf{o}^o\|^2 + 2\mu tr\left(\breve{G}U^*\mathbf{o}^o\bar{\mathbf{x}}^{o^T}\left(I_N - \breve{G}\right)\right)$$
$$+ 2tr\left(\breve{\mathcal{B}}\mathbb{E}\tilde{\mathbf{x}}_{i-1}\left(\left(I_N - \breve{G}\right)\bar{\mathbf{x}}^o - \mu\breve{G}U^*\mathbf{o}^o\right)\right) \tag{4.48}$$

Comparing the expressions in (4.48) and (4.46), we rewrite Eq.(4.46) as

$$\mathbb{E}\|\tilde{\mathbf{x}}_i\|^2 = \mathbb{E}\|\breve{\mathcal{B}}_i\tilde{\mathbf{x}}_{i-1}\|^2 + \|\mathbb{E}\tilde{\mathbf{x}}_i\|^2 + \|\breve{\mathcal{B}}\mathbb{E}\tilde{\mathbf{x}}_i\|^2 + tr\left(\breve{\mathcal{B}}\mathbb{E}\left(\mathbf{q}_i\mathbf{q}_i^T\right)\breve{\mathcal{B}}^T\right) + \mu^2 tr\left(\breve{G}R_u\mathbb{E}\left(\mathbf{e}_i\mathbf{e}_i^T\right)\breve{G}\right)$$
$$+ tr\left(\left(I_N - \breve{G}\right)\mathbb{E}\left(\left(\mathbf{x}_i^o - \bar{\mathbf{x}}^o\right)\left(\mathbf{x}_i^o - \bar{\mathbf{x}}^o\right)^T\right)\left(I_N - \breve{G}\right)^T\right) \tag{4.49}$$

We assume $\mathbb{E}\breve{G}_i\mathbf{q}_j = \mathbf{0}$ for any $i \neq j$ and from the true signal model in (2.34), we have

$$\left(I_N - \breve{G}\right)\mathbb{E}\left(\left(\mathbf{x}_i^o - \bar{\mathbf{x}}^o\right)\left(\mathbf{x}_i^o - \bar{\mathbf{x}}^o\right)^T\right)\left(I_N - \breve{G}\right) = \left(I_N - \breve{G}\right)\mathbb{E}\left(\mathbf{q}_i\mathbf{q}_i^T\right)\left(I_N - \breve{G}\right) \tag{4.50}$$

Then, we get the steady state the mean-square error from (4.46) and (4.47) as,

$$\mathbb{E}\|\tilde{\mathbf{x}}_\infty\|^2 = \lim_{i\to\infty}\mathbb{E}\|\tilde{\mathbf{x}}_i\|^2$$
$$\approx \underbrace{\|\mathbb{E}\tilde{\mathbf{x}}_\infty\|^2}_{\text{bias squared.}} + \underbrace{c\ tr\left(\sum_{j=0}^{\infty}\breve{B}^j\left(I_N - \mu\breve{G}R_u\right)L^\dagger\left(I_N - \mu\breve{G}R_u\right)^T\breve{B}^{j^T}\right)}_{\text{error due to the non-stationary data model.}}$$
$$+ \underbrace{\mu^2 tr\left(\sum_{j=0}^{\infty}\breve{B}^j\breve{G}R_u\Sigma\breve{G}^T\breve{B}^{j^T}\right)}_{\text{error due to the measurement noise.}}, \tag{4.51}$$

To find the mean-squared error of $\tilde{\mathbf{o}}_i$, first, we write the square of $\mathbb{E}\tilde{\mathbf{o}}_i$ as

$$\|\mathbb{E}\tilde{\mathbf{o}}_i\|^2 = (1-\mu)^2\|\mathbb{E}\tilde{\mathbf{o}}_{i-1}\|^2 + \mu^2 tr\left(R_u\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)^T\right) - 2(1-\mu)\mu tr\left(U\mathbb{E}\tilde{\mathbf{x}}_{i-1}\mathbb{E}\tilde{\mathbf{o}}_{i-1}^T\right) \tag{4.52}$$

Comparing the expressions in (4.47) and (4.52), we rewrite (4.47) as

$$\mathbb{E}\|\tilde{\mathbf{o}}_i\|^2 = (1-\mu)^2\mathbb{E}\|\tilde{\mathbf{o}}_{i-1}\|^2 + \mu^2 tr\left(\Sigma\right) + \|\mathbb{E}\tilde{\mathbf{o}}_i\|^2 - (1-\mu)^2\|\mathbb{E}\tilde{\mathbf{o}}_{i-1}\|^2$$
$$+ \mu tr\left(R_u\mathbb{E}\left(\tilde{\mathbf{x}}_{i-1}\tilde{\mathbf{x}}_{i-1}^T\right) - U\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}\right)\left(\mathbb{E}\tilde{\mathbf{x}}_{i-1}^T\right)U^*\right) \tag{4.53}$$

Then in steady state,

$$\begin{aligned}\mathbb{E}\|\tilde{\mathbf{o}}_\infty\|^2 &= \lim_{i\to\infty}\mathbb{E}\|\tilde{\mathbf{o}}_i\|^2 \\ &\approx \|\mathbb{E}\tilde{\mathbf{o}}_\infty\|^2 + \frac{\mu^2}{1-(1-\mu)^2}tr\left(\Sigma\right) \\ &\quad + \frac{1}{1-(1-\mu)^2}tr\left(R_u\mathbb{E}\left(\tilde{\mathbf{x}}_\infty\tilde{\mathbf{x}}_\infty^T\right) - U\left(\mathbb{E}\tilde{\mathbf{x}}_\infty\right)\left(\mathbb{E}\tilde{\mathbf{x}}_\infty^T\right)U^*\right) \end{aligned} \tag{4.54}$$

Let $MSD_x^l$ and $MSD_o^l$ be the network mean square deviation for the estimator $\mathbf{x}$ and $\mathbf{o}$, respectively. Then from (4.51) and (4.54), we conclude that

$$MSD_x^l = \frac{1}{N}\mathbb{E}\|\mathbf{x}_\infty\|^2 < \infty, \tag{4.55}$$

$$MSD_o^l = \frac{1}{N}\mathbb{E}\|\mathbf{o}_\infty\|^2 < \infty. \tag{4.56}$$

## 4.4 Experiments

### 4.4.1 Experiment on Synthetic Data

We consider an undirected network with 100 nodes randomly distributed over a plane. We form a network Laplacian $L$ based on the Euclidean distance between the nodes. First, we generate the initial network signal $\mathbf{x}_i^o$ from a Gaussian of mean 10 and the covariance $L^\dagger$. We generate the non-stationary ground truth $\mathbf{x}_i^o$ using (2.4). Each node $k$ has the maximum $N_k$ of 6. All the experiments are run with the Signal-to-Noise ration of 0 dB and the injected outlier values are randomly selected within the variance of the measurement noise. We set $a_{l,k}(i) = \frac{1}{N_k}$ and $\mu_k = \mu_k' = 0.1$ and $\{\alpha_k', \gamma_k\}$ are set to 0.5. We analyze the performance of the proposed algorithm under three scenarios; a growing cluster outlier in a static network, randomly distributed burst outliers in a static network and time-varying network. In all cases, we compare

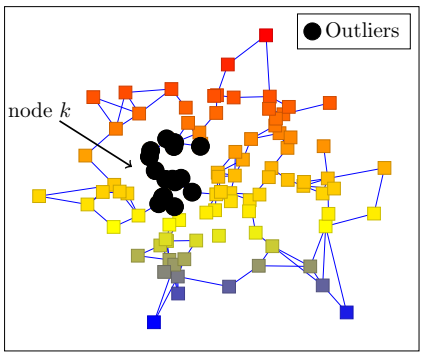
the MSD performances between the DLMS-LGT and outliers estimation in Alg.6 and the distributed Laplacian regularized LMS with the traditional local penalty function where the regularized parameter are designed using [60] and [57] denoted as Dist.LR-LMS. All the simulations are averaged over 100 run.

**Static Network with Growing Cluster Outliers**

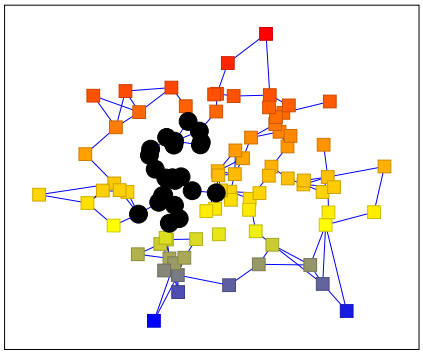
In this experiment, we test the performance of the proposed algorithm in a static network with growing outlier neighborhood at time $i = 0, 800, 1600$, respectively. We randomly select one node $k$ and initially add the random constant outlier signals to node $k$ and its 3-hop neighbors. At $i = 800$, we inject the random constant outlier signal to node $k$ and its 4-hop neighbors and at $i = 1600$ to 5-hop neighbors as shown in Figure 4.3. The non-stationary true network signal are generated as mentioned above. Figure 4.4 shows the MSDs comparison for the estimated graph signal and outlier signal. As we can see, the proposed LGT algorithm is less sensitive to the cluster outliers in the network and converges to the theoretical MSD. However, the global/centralized algorithm (GGF) is sensitive to the cluster outliers and its performance declines as the size of outlier neighborhood increases.

**Static Network with Non-Stationary Signal and Outliers**

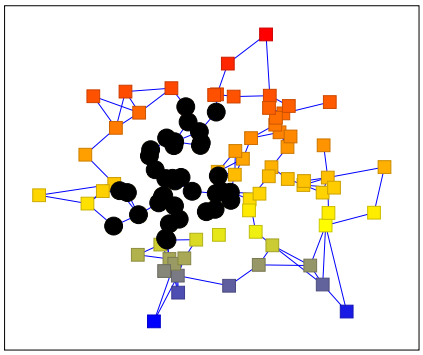
Here, we test the proposed LGT and outlier estimation algorithm for randomly selected burst outliers in a static network. We randomly select 5 nodes and inject the burst outlier signals throughout the simulation. The magnitude of each burst are randomly selected for the value less than the variance of the measurement noise. Each burst arrival time is drawn from the Poisson distribution with the average rate of 1 per




(a) Iteration $i = 0$



(b) Iteration $i = 800$



(c) Iteration $i = 1600$

Figure 4.3: Network topology with growing cluster outliers. The color on nodes represented by the square marker represent the true graph signal. The random constant outliers are injected to node $k$ and (a) its 3-hop neighbors, (b) 4-hop neighbors and (c) 5-hop neighbors at time $i = 0, 800$ and 1600, respectively.

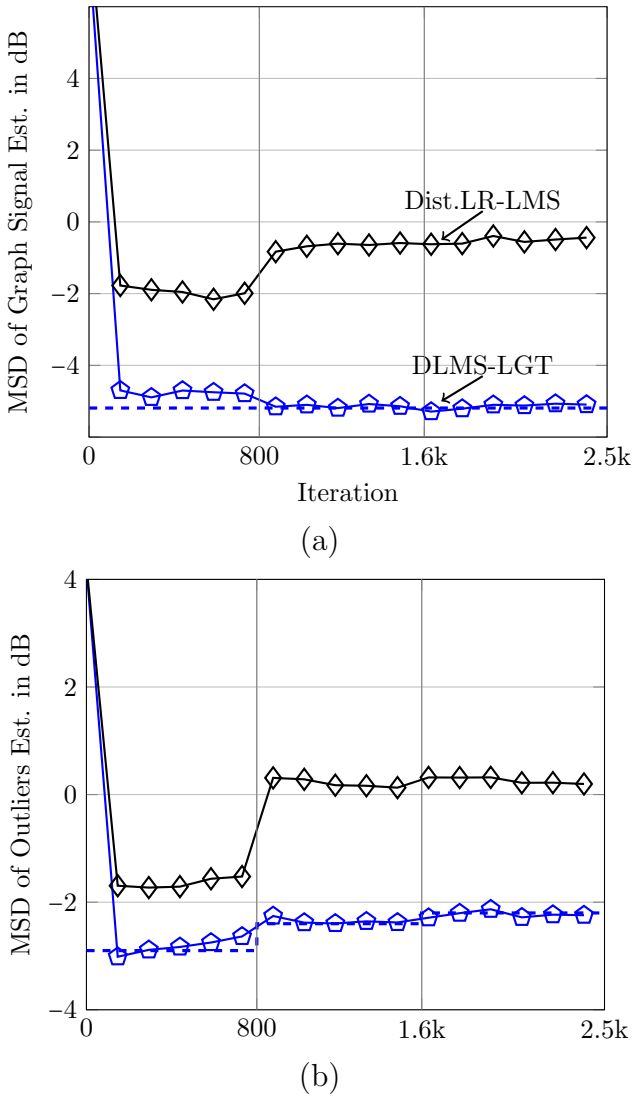



Figure 4.4: MSD comparison of (a) graph signal and (b) outlier with growing cluster outliers in a static network shown in Figure 4.3. The dashed line represents the theoritical MSD value of DLMS-LGT-outlier calculated using (4.55)

.

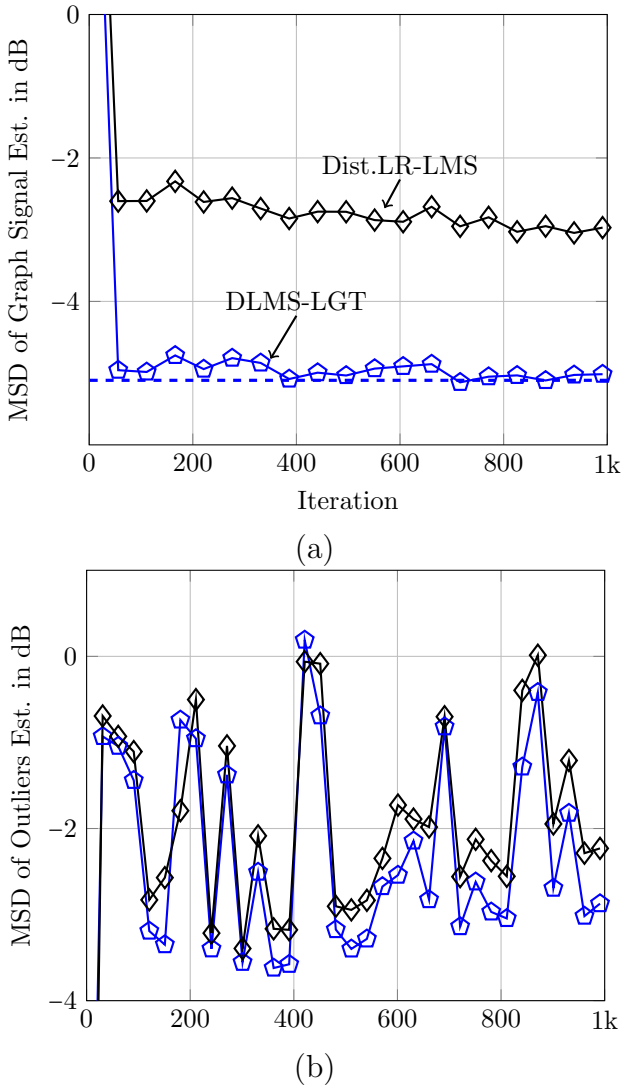



Figure 4.5: MSD comparison of the estimated graph signal (a) and the outliers (b) in a static network with the bursh outliers. The dashed line represents the theoretical MSD value of the DLMS-LGT algorithm calculated using Eq.(4.55) conditioning on the perfect information of the outlier.

every 100 iterations and the burst length is also drawn from the Poisson distribution with the average length of 50 iterations. Figure 4.6 shows the outlier tracking ability of the DLMS-LGT Alg.6. Figure 4.5 have the MSD comparison between DLMS-LGT and Dist.LR-LMS. As we can see the DLMS-LGT approach performs well in tracking the non-stationary signal and estimating the outliers, with the LGT-based estimation providing the lowest MSD for both tasks. Moreover, we see that the DLMS-LGT empirical MSD for tracking the non-stationary signal roughly follows the theoretical MSD, whose plot is obtained by conditioning on perfect knowledge of the outliers.

**Time-Varying Network**

Now, we simulate the proposed algorithm in time-varying networks. As before, we randomly select 5 nodes and generate the sequences of burst outliers as mentioned above. For the time-varying network, we change 10% of the total edges at iteration 330 and 690, and so after each change $\mathbf{q}_i$ in Eq.(2.4) is drawn from the new $L^{\dagger}$. Figure 4.7 (a) and (b) show the MSD comparison between the three algorithms for the graph signal and the outlier estimation, respectively. As we can see the DLMS-LGT method follows the theoretical MSD using Eq.(4.55) that are once again conditioned on perfect knowledge of the outliers, and calculated as though there will be no network changes. As we can see the LGT-based method outperforms the other two algorithms in estimating the outliers signal as well. We also compare the detection rate of the algorithms. The Generalized Log- Likelihood Ratio Test (GLRT) is performed on every 25 iterations. Figure 4.8 shows the ROC comparison of the burst outliers detection in time-varying network between the three algorithms.

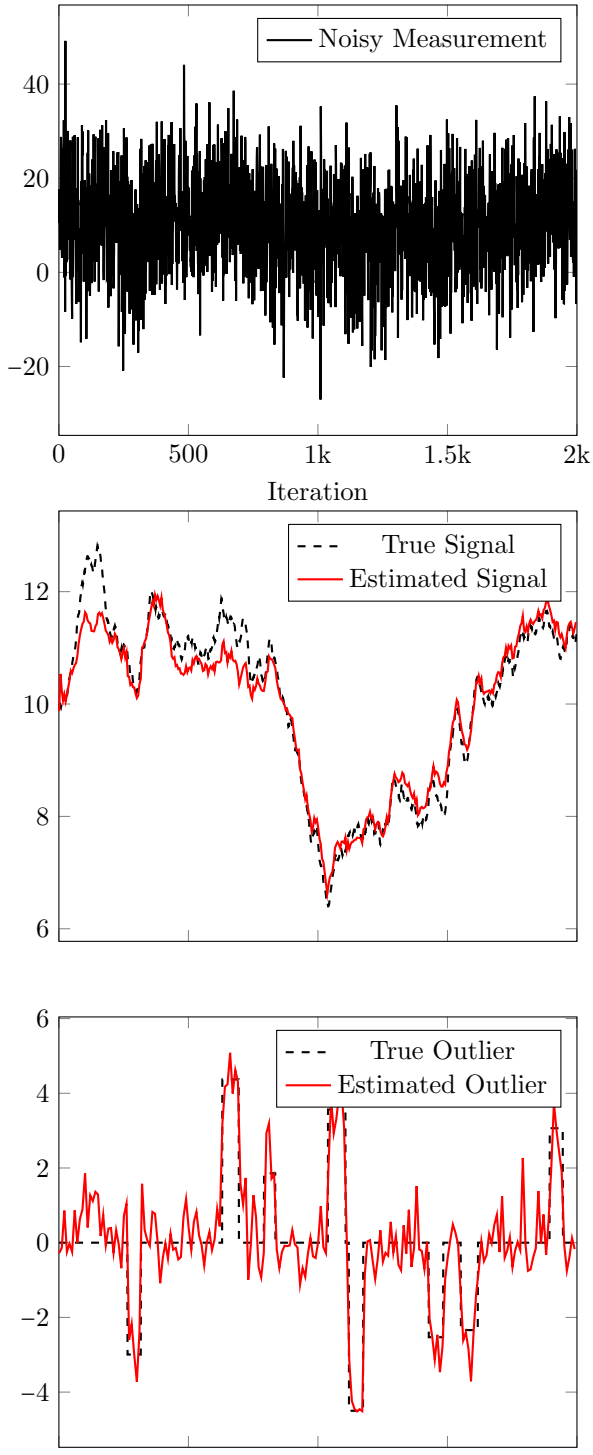



Figure 4.6: Tracking Performance of DLMS-LGT-Outliers Alg.6 for non-stationary signal and the burst outliers.

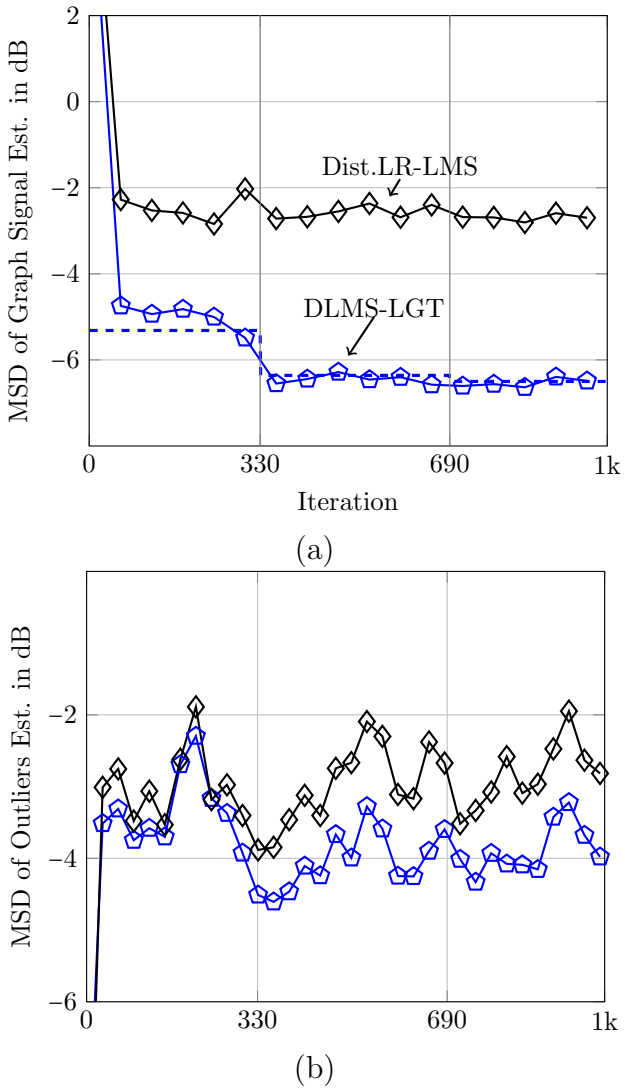



Figure 4.7: MSD comparison of the estimated graph signal (a) and the outliers (b), respectively.The dashed line represents the theoretical MSD value of the DLMS-LGT-outlier algorithm calculated using Eq.(4.55) conditioning on the perfect information of the outliers.

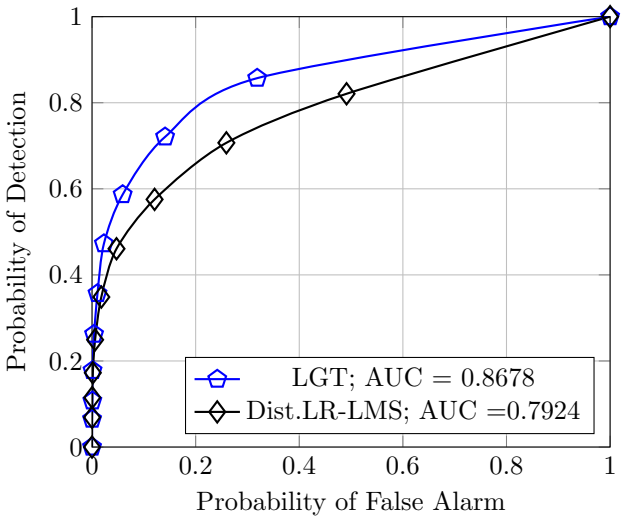



Figure 4.8: ROC comparison of the non-stationary burst outliers detection in time-varying networks.

### 4.4.2 Experiment on Real-World Dataset

**Weather Data, [65]**

We evaluate the proposed DLMS-LGT and outliers estimation on Weather Data [65]. We selected 130 stations across the United States and form a graph based on the geodesic distance between the stations. The average number of neighbors is limited to 6. The station topology is shown in Figure 4.9 (a). We randomly selected 30% of the stations and added the burst outliers throughout the experiments. The arrival time and the length of the outliers are drawn from the Poisson distribution with the average arrival of 1 arrival per 150 iteration and average burst length of 50. Figure 4.9 (c) and (d) have the MSD comparison of the true temperature and outliers estimation between the algorithms. In Figure 4.9 (b), we provide the ROC comparison of the outliers detection. We run the window GLRT on the estimated outliers with the

window size of 25. As we can see the detection rate of the DLMS-LGT-based method outperforms the traditional algorithm.

**Abilene Backbone Network, [66]**

Next, we evaluate our algorithm using the Abilene backbone network traffic dataset [66]. The backbone network has 30 edges. We consider the edges in the Abilene network as the nodes in our graph and we form the connection between the two nodes if the two nodes (two edges in Abilene network) are connected. Then, we injected the burst outliers to the benign traffic. Again the outliers are drawn from Poisson distribution with the average arrival time of 1 per 500 iteration and average burst length of 300. Then, we run the window GLRT on the estimated outliers with the window size of 150 and the ROC results are shown in Figure 4.19. Figure 4.22 shows the tracking performance of the DLMS-LGT algorithm on benign traffic and outliers estimation.

**UGR'16 Traffic Dataset, [67]**

We randomly selected 300 IPs from the traffic data reported from 07/27/2016 to 07/31/2016 from UGR'16 Dataset, [67]. First, we take 50% of the data to learn the network Laplacian following the graph learning methods from [62, 68–71]. We injected the outliers to the outgoing packages and we run the DLMS-LGT and outleirs estimation at each IP which has outgoing packages at the instant $i$. The outliers are drawn from Poisson distribution with the average arrival time of 1 per 50 iteration and average burst length of 60. We also perform the GLRT to the estimated outliers at each IP with the window size of 30. The ROC comparison between the algorithms

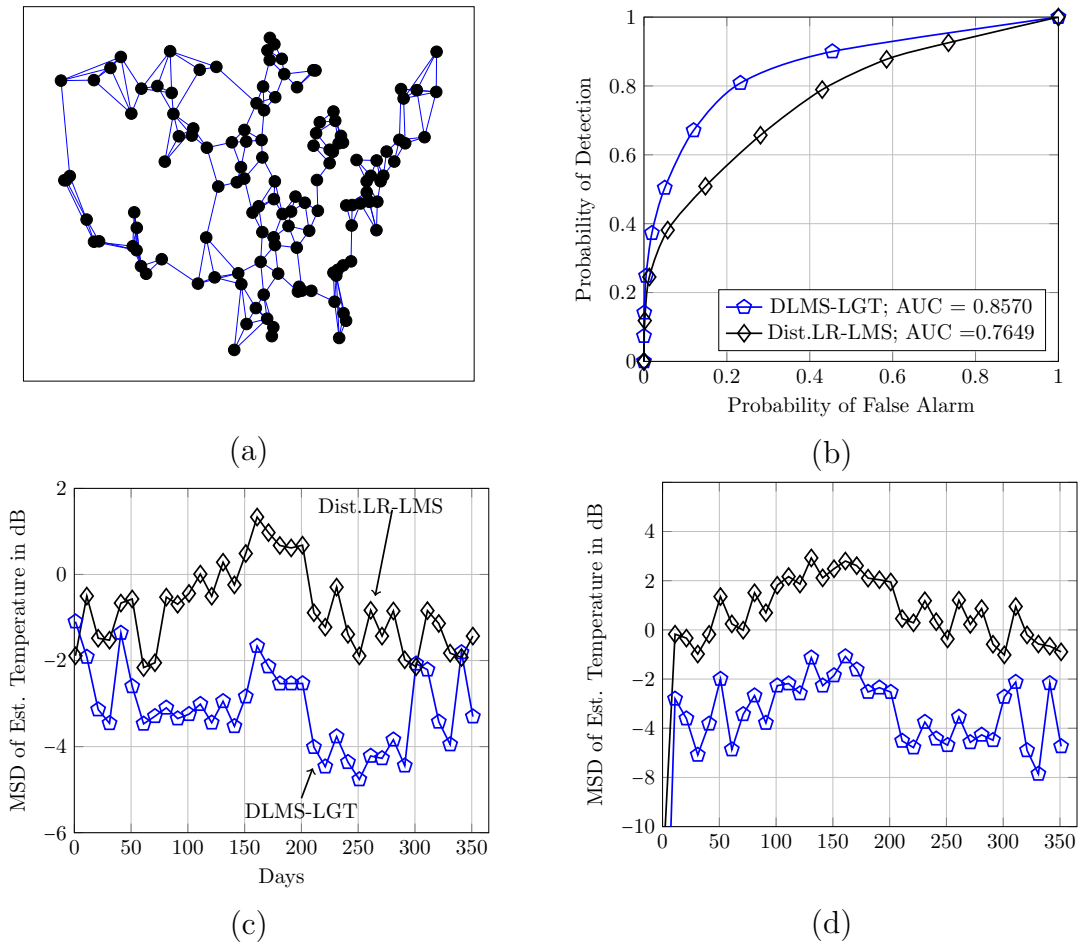



Figure 4.9: (a) Network Topology of 130 Weather Stations across United States, [65]. The color represents the temperature distribution across the stations. The true temperature values are corrupted with the zero-mean Gaussina noise $\sigma$ = 1 and the burst outliers. The (b) ROC comparison of the algorithms for detection the burst outliers of the weather station network. (c) and (d) MSD Comparison of the true temperature value and the burst outliers estimation of the algorithms.

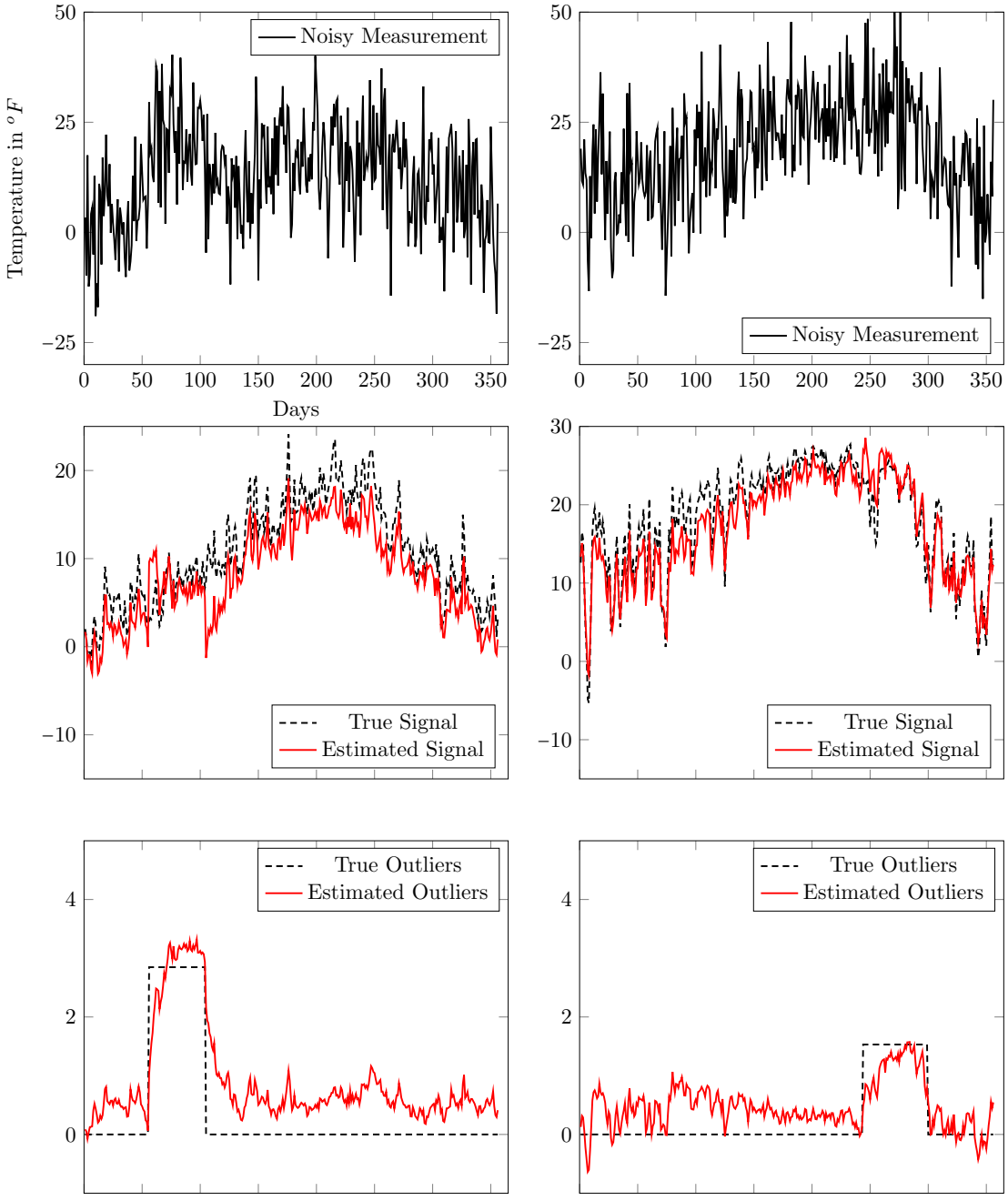



Figure 4.10: Tracking Performance of DLMS-LGT-Outlier Alg. 6 of Weather Dataset, [65].

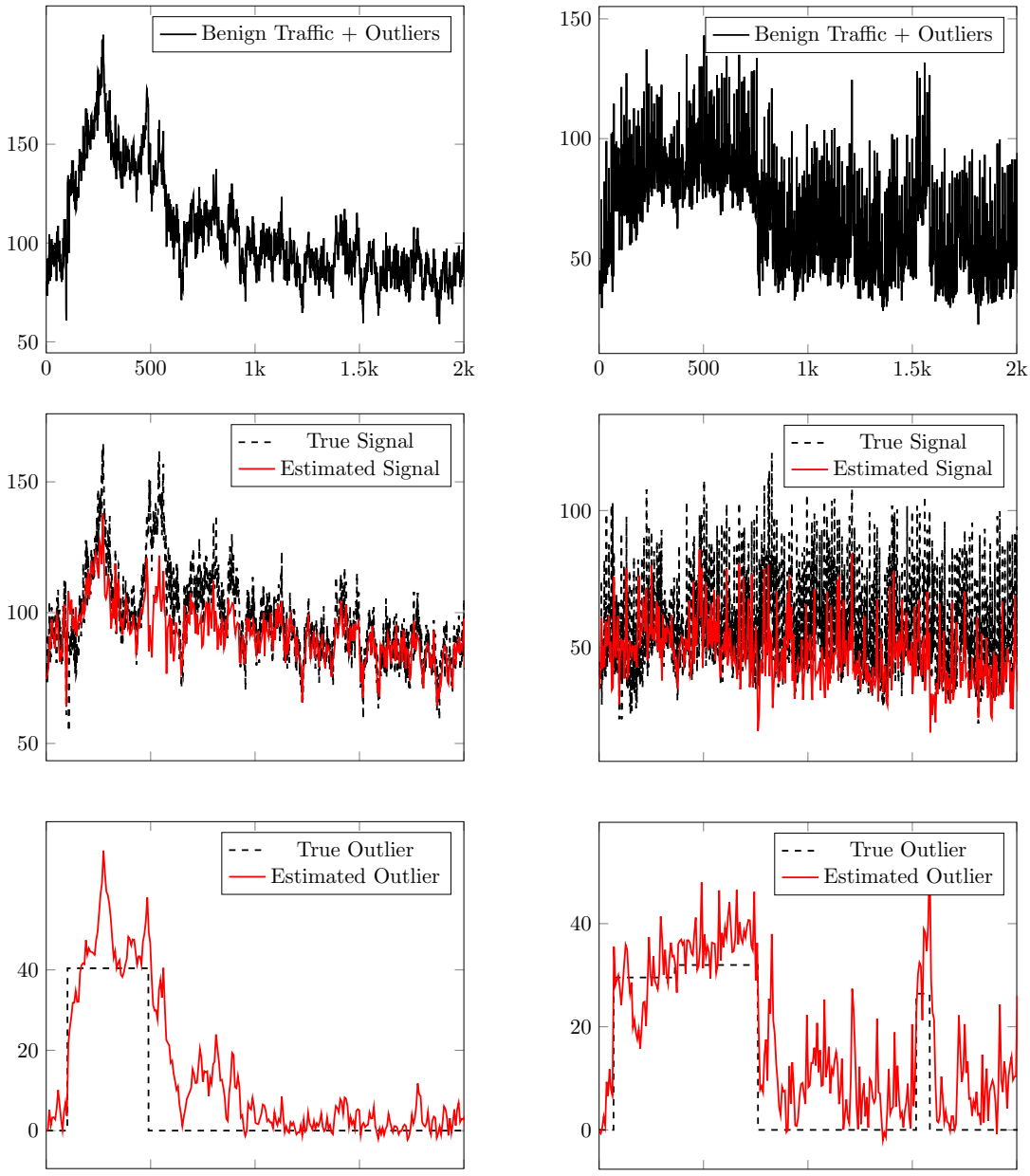



Figure 4.11: Tracking Performance of DLMS-LGT-Outlier on Backbone Network Dataset.

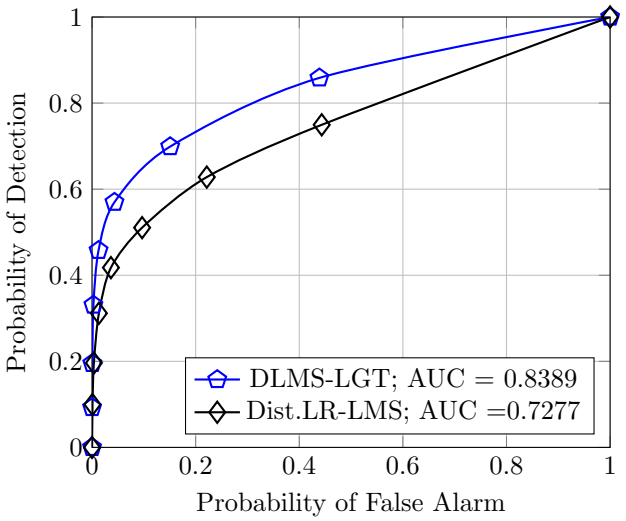



Figure 4.12: ROC Comparison on Abilene Backbone Network, [66].

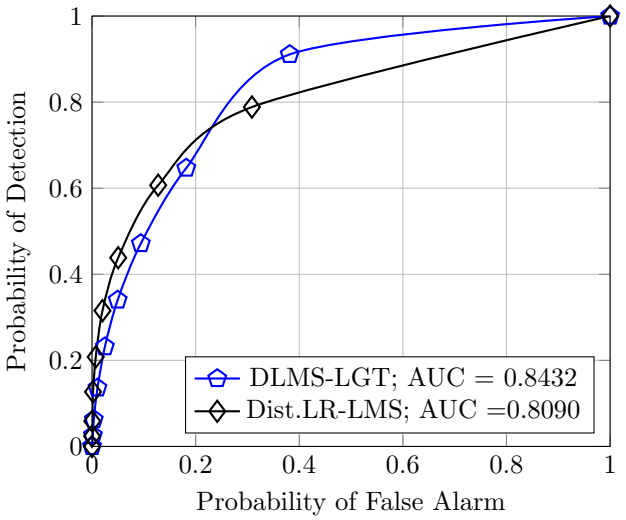



Figure 4.13: ROC Comparison of the burst outliers detection on UGR'16 Dataset, [67].

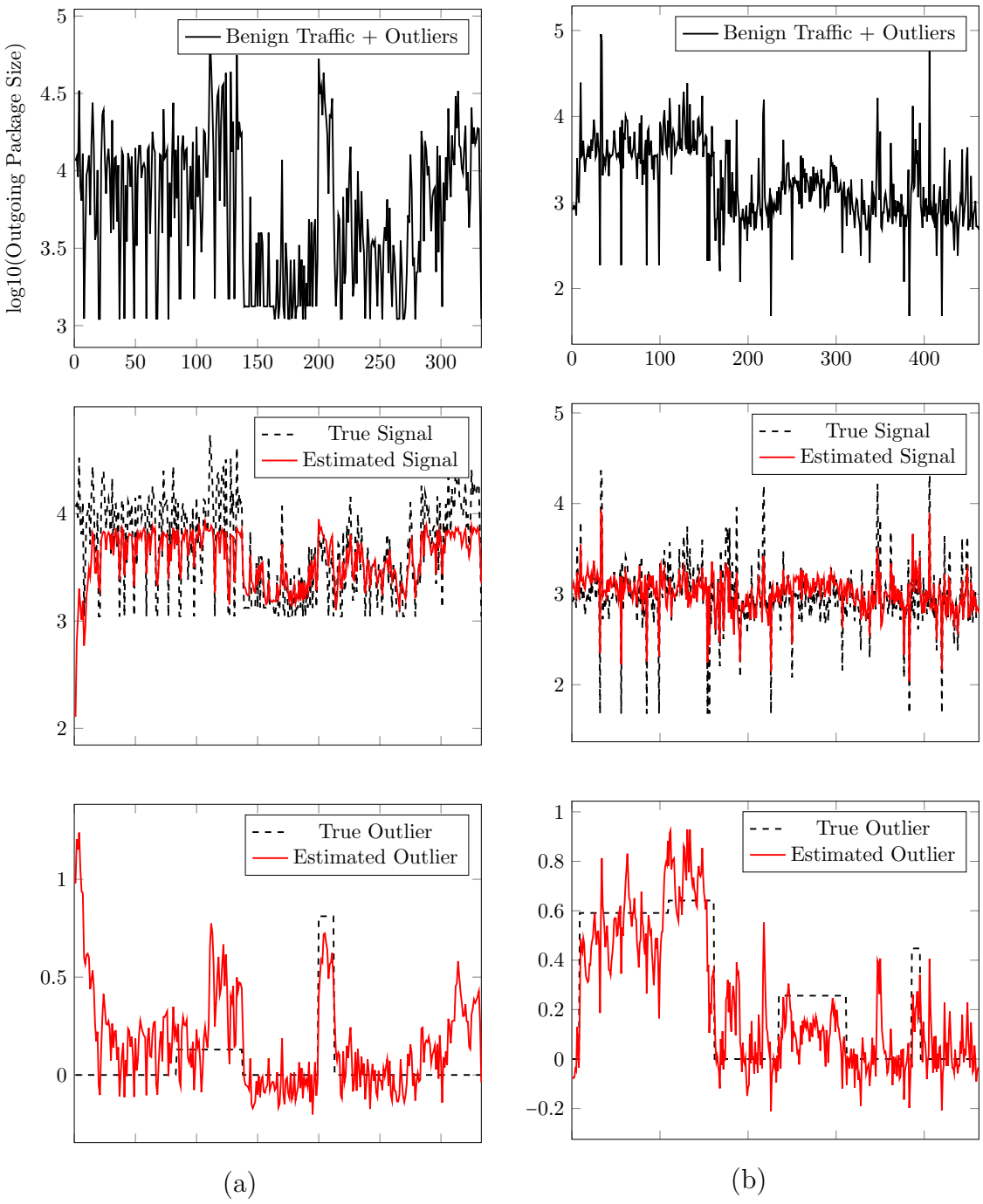



Figure 4.14: Tracking Performance of DLMS-LGT-Outlier on UGR'16 Dataset.

are shown in Figure 4.13. Figure 4.14 shows the performance of the DLMS-LGT for tracking the benign traffic and the injected outliers.

## 4.5 Signal and Outliers Estimation in Local Graph with LS Strategy

In previous section, we studied the estimating and tracking of non-stationary signal and outliers in graph with the distributed least-mean square strategy and the LGT filters. Here, we will explore the graph signal and outliers estimation using distributed Least Square strategy incorporating with the LGT filters.

Consider each node $k$ has a noisy measurement data $y_k(i)$ for the true signal $x_k^o(i)$ and $o_k^o(i)$ is a possible outlier signal at node $k$. We define the network measurement model as

$$\mathbf{y}_i = \mathbf{x}_i^o + \mathbf{o}_i^o + \mathbf{e}_i \tag{4.57}$$

where $\mathbf{e}_i$ is a measurement noise and $\mathbf{y}_i = \{y_1(i), \ldots, y_N(i)\}_{N\times 1}$

$\mathbf{x}_i^o = \{x_1^o(i), \ldots, x_N^o(i)\}_{N\times 1}$ and

$\mathbf{o}_i^o = \{o_1^o(i), \ldots, o_N^o(i)\}_{N\times 1}$.Let the graph signal $\mathbf{x}_i^o$ evolves in time as, [1],

$$\mathbf{x}_i^o = a\mathbf{x}_{i-1}^o + \mathbf{q}_i \tag{4.58}$$

where $|a| < 1$ and $\mathbf{q}_i$ is some random perturbation such as $\mathbf{q}_i \sim \mathcal{N}(\bar{\mathbf{q}}, L^\dagger)$ in which $L$ is the normalized global Laplacian of the network. With the signal model in (4.58), we assume the graph signal $\mathbf{x}_i^o$ is defined over graph Laplacian $L$. The oultiers signal $\mathbf{o}_i^o$ is assumed to be sparse and varying in time. Hence, we introduce the global cost

for estimating the graph signal $\mathbf{x}_i^o$ and the outliers $\mathbf{o}_i^o$ as

$$J^g\left(\boldsymbol{\psi}_i, \mathbf{x}_i, \mathbf{o}_i, \mathbf{z}_i\right) = \mathbb{E}\|\mathbf{y}_i - \boldsymbol{\psi}_i - \mathbf{o}_i\|^2 + \sum_{j=0}^{i} r^{i-j}\|\boldsymbol{\psi}_i - \mathbf{x}_i\|^2 + \beta \mathbf{x}_i^T L \mathbf{x}_i + \alpha\|\mathbf{o}_i\|_1 + \gamma \mathbf{z}_i^T\left(\mathbf{y}_i - \boldsymbol{\psi}_i - \mathbf{o}_i\right) \tag{4.59}$$

where $\boldsymbol{\psi}_i$ and $\mathbf{z}_i$ are the auxiliary variables and $\beta$,$\alpha$ and $\gamma$ are the non-negative coefficients. We add the penalty cost $\mathbf{x}_i^T L \mathbf{x}_i$ in (4.59) in order to minimize the total variation of the estimated signal $\mathbf{x}_i$ across the network. We are interested to find the distributed solution for the global cost in (4.59). We start by writing the global cost (4.59) as a sum of each node $k$'s local cost. To begin, we introduce an $N_k \times N_k$ local Laplacian at each node $k$; $S_k$. We introduce $\psi_k$ an intermediate variable at each node $k$ and an $N_k \times 1$ vector $\boldsymbol{\psi}_k$ contains $\psi_\ell$ for $\ell \in \mathcal{N}_k$ with $\psi_k$ as the first element in $\boldsymbol{\psi}_k$. Note that node $k$ in inclusive in $\mathcal{N}_k$. Using the local Laplacian $S_k$ from Section 3.1 we introduce each node $k$'s local cost as

$$J_k^{dist}\left(\psi_{k,i}, \mathbf{x}_{k,i}, o_k(i), z_k\right) = \mathbb{E}|y_k(i) - \psi_{k,i} - o_k(i)|^2 + \sum_{j=0}^{i} r^{i-j} \sum_{\ell \in \mathcal{N}_k} \|P_k^T P_\ell \boldsymbol{\psi}_{\ell,i} - \mathbf{x}_{k,i}\|^2 + \beta_k \mathbf{x}_{k,i}^T S_k \mathbf{x}_{k,i} + \sum_{\ell \in \mathcal{N}_k} a_{\ell,k}\|P_k \mathbf{x}_{k,i} - P_\ell \mathbf{x}_{\ell,i}\|^2 + \alpha|o_k(i)| + \gamma_k z_k\left(y_k(i) - \psi_k(i) - o_k(i)\right) \tag{4.60}$$

Note that $o_k(i)$ is the estimate of $o_k^o(i)$ and $\mathbf{x}_{k,i}$ is the estimate of the true signal of node $k$ and its neighbors; $\mathbf{x}_{k,i}^o = P_k \mathbf{x}_i^o$ and $P_k$ is an $N_k \times N$ permutation matrix. Using the ADMM principle [63, 64], each node $k$ iteratively solves the above cost function

as follows;

$$\psi_k(i) = \min_{\psi_k(i)} J_k^{dist}\left(\psi_k(i), \mathbf{x}_{k,i-1}, o_k(i-1), z_k(i-1)\right) \tag{4.61}$$

$$\mathbf{x}_{k,i} = \min_{\mathbf{x}_k} J_k^{dist}\left(\psi_k(i), \mathbf{x}_k, o_k(i-1), z_k(i-1)\right) \tag{4.62}$$

$$o_k(i) = \min_{o_k(i)} J_k^{dist}\left(\psi_k(i), \mathbf{x}_{k,i}, o_k(i), z_k(i-1)\right) \tag{4.63}$$

$$z_k(i) = \min_{z_k} J_k^{dist}\left(\psi_k(i), \mathbf{x}_{k,i}, o_k(i), z_k(i-1)\right). \tag{4.64}$$

We start with finding $\boldsymbol{\psi}_{k,i}$ as

$$\psi_k(i) = \frac{1}{1+\eta_k}\Big(y_k(i) - o_k(i-1) + \gamma_k z_k(i-1) + \eta_k \sum_{j=0}^{i-1} r^{i-j-1} x_k(j)\Big). \tag{4.65}$$

where $x_k(j)$ is the first element of the vector $\mathbf{x}_{k,j}$. Let

$$\begin{aligned}\varphi_k(i) &= \sum_{j=0}^{i} r^{i-j} x_k(j) \\ &= r\varphi_k(i-1) + x_k(i)\end{aligned} \tag{4.66}$$

Rewrite (4.65) as

$$\psi_k(i) = \frac{1}{1+\eta_k}\Big(y_k(i) - o_k(i-1) + \gamma_k z_k(i-1) + \eta_k \varphi_k(i-1)\Big). \tag{4.67}$$

Node $k$ exchanges $\psi_k(i)$ with neighbors and form the vector $\boldsymbol{\psi}_{k,i}$. Then, node $k$ solves (4.62) as

$$\boldsymbol{\phi}_{k,i} = \frac{1}{1+r_k} \underbrace{\left(I_k + \ddot{G}_{k,i}\right)^{-1}}_{\text{DLS-LGT filter.}} \boldsymbol{\psi}_{k,i} + \frac{r_k}{1+r_k}\mathbf{x}_{k,i-1} \tag{4.68}$$

$$\mathbf{x}_{k,i} = \sum_{\ell \in N_k} a_{\ell,k} P_k^T P_\ell \boldsymbol{\phi}_{k,i}. \tag{4.69}$$

We denote $\left(I_k + \ddot{G}_{k,i}\right)^{-1}$ as node $k$'s LGT filter and $\ddot{G}_{k,i}$ is node $k$'s local graph operator; $\ddot{G}_{k,i} = V_k \ddot{\Theta}_{k,i} V_k^T$ and $\ddot{\Theta}_{k,i}$ is node $k$'s filter response matrix where $\ddot{\Theta}_{k,i} =$

$diag\{\ddot{\theta}_{k,1}(i),\ldots,\ddot{\theta}_{k,N_k}(i)\}$. Let $\mathbf{f}^o_{k,i}$ and $\mathbf{\Psi}_{k,i}$ be the frequency representation of $\mathbf{x}^o_{k,i}$ and $\boldsymbol{\psi}_{k,i}$ in node $k$'s LGT, respectively, $\mathbf{f}^o_{k,i} = V_k^T\mathbf{x}^o_{k,i}$ and $\mathbf{\Psi}_{k,i} = V_k^T\boldsymbol{\psi}_{k,i}$.

$$\min_{\ddot{\theta}_{k,n}} \|\mathbf{f}^o_{k,i}(\lambda_{k,n}) - \left(1+\ddot{\theta}_{k,n}\right)^{-1}\mathbf{\Psi}_{k,i}(\lambda_{k,n})\|^2, \forall\lambda_{k,n}, \forall k. \tag{4.70}$$

where $\lambda_{k,n}$ is the frequency of node $k$'s LGT and $\mathbf{f}^o_{k,i}(\lambda_{k,n})$ and $\mathbf{\Psi}_{k,i}(\lambda_{k,n})$ are the $n^{th}$ element of the vector $\mathbf{f}^o_{k,i}$ and $\mathbf{\Psi}_{k,i}$, respectively. As in Eq.(3.42), the filter response $\ddot{\theta}_{k,n}(i)$ is calculated as in (3.42),

$$\ddot{\theta}_{k,n}(i) = \max\{0, \frac{\mathbf{\Psi}_{k,i}(\lambda_{k,n})^2}{2\zeta_{k,n}} - \frac{1}{2}\} \leq \ddot{\theta}_{max}. \tag{4.71}$$

Then, from [56], each node $k$ iteratively estimates $o_k(i)$ as follows;

$$o_k(i) = (1-\mu_k)o_k(i-1) + \mu_k(y_k(i) - \psi_k(i) + \gamma_k z_k(i-1) - \alpha'_k sgn\{o_k(i-1)\} \tag{4.72}$$

where $\mu_k$ is a non-negative step-size and $\alpha'_k = \mu_k\alpha_k$. Update the auxiliary variable $z_k$;

$$z_k(i) = z_k(i-1) + \gamma_k\left(y_k(i) - \psi_k(i) - o_k(i)\right). \tag{4.73}$$

The complete algorithm of the distributed Least square with the LGT and outliers estimation is given in Algorithm (7).

## 4.6 Performance of DLS-LGT Outliers

### 4.6.1 Stability and Mean Convergence

Now, let us study the behavior of the proposed DLS-LGT algorithm. Let $\mathbf{y}_{k,i} = \{y_k(i), 0, \ldots, 0\}_{N_k\times 1}$,

$\mathbf{o}_{k,i} = \{o_k(i), 0, \ldots, 0\}_{N_k\times 1}$, and

**Algorithm 7** DLS-LGT and Outliers Estimation

Initialize $\alpha_0 = 1$, $o_k(0) = 0$, $z_k(i) = 0$, $\psi_k(0) = 0$ for all $k$ and node $k$ has $\zeta_{k,n}, n \in \{1, \ldots N_k\}$.

**For** $i > 0$ and $k = 1$ to $N$ **do**

1. Each node $k$ has $y_k(i)$.
2. $\psi_k(i) = \frac{1}{1+\eta_k}\left(y_k(i) - o_k(i-1) + \gamma_k z_k(i-1) + \eta_k \varphi_k(i-1)\right)$.
3. Exchange $\psi_k(i)$ with the neighbors and form $\boldsymbol{\psi}_{k,i}$.
4. Find the LGT filter response.

   **For** $n = 1$ to $N_k$ **do**

   $\ddot{\theta}_{k,n}(i) = \max\{0, \frac{\boldsymbol{\Psi}_{k,i}(\lambda_{k,n})^2}{2\zeta_{k,n}} - 1\} \leq \ddot{\theta}_{max}$

   **End.**

   $\ddot{G}_{k,i} = V_k \ddot{\Theta}_{k,i} V_k^T$ where $\ddot{\Theta}_{k,i} = diag\{\ddot{\theta}_{k,1}(i), \ldots, \ddot{\theta}_{k,N_k}(i)\}$.
5. $\boldsymbol{\phi}_{k,i} = \frac{1}{1+r_k}\left(I_k + \ddot{G}_{k,i}\right)^{-1} \boldsymbol{\psi}_{k,i} + \frac{r_k}{1+r_k}\mathbf{x}_{k,i-1}$
6. Exchange $\boldsymbol{\phi}_{k,i}$ with the neighbors.
7. $\mathbf{x}_{k,i} = \sum_{\ell \in N_k} a_{\ell,k} P_k^T P_\ell \boldsymbol{\phi}_{k,i}$.
8. Update $\varphi_k(i) = r\varphi_k(i-1) + x_k(i)$.

   $x_k(i)$ is the first element of $\mathbf{x}_{k,i}$.
9. $o_k(i) = (1-\mu_k) o_k(i-1) + \mu_k\left(y_k(i) - \psi_k(i) + \gamma_k z_k(i)\right) - \alpha'_k sgn\{o_k(i-1)\}$.
10. $z_k(i) = z_k(i-1) + \gamma_k(y_k(i) - \psi_k(i) - o_k(i))$.

**End.**

$\mathbf{z}_{k,i} = \{z_k(i), 0, \ldots, 0\}_{N_k \times 1}$. We express Eq.(4.68) and (4.69) in a recursion as,

$$\begin{aligned}\mathbf{x}_{k,i} &= \sum_{\ell \in \mathcal{N}_k} \frac{a_{\ell,k}}{1+r_\ell} P_k^T P_\ell \Big( \big(I_\ell + \ddot{G}_{\ell,i}\big)^{-1} \boldsymbol{\psi}_{\ell,i} + r_\ell \mathbf{x}_{\ell,i} \Big) \\ &= \sum_{\ell \in \mathcal{N}_k} \frac{a_{\ell,k}}{1+r_\ell} P_k^T P_\ell \Big( \frac{1}{1+\eta_\ell} \big(I_\ell + \ddot{G}_{\ell,i}\big)^{-1} \\ &\quad . \sum_{n \in \mathcal{N}_\ell} P_\ell^T P_n \left(\mathbf{y}_{n,i} - \mathbf{o}_{n,i-1} + \gamma_n \mathbf{z}_{n,i-1} + \eta_n \mathbf{x}_{n,i-1}\right) + r_\ell \mathbf{x}_{\ell,i-1} \Big). \end{aligned} \tag{4.74}$$

For simplicity, $\psi_k(i) = \frac{1}{1+\eta_k}\left(y_k(i) - o_k(i-1) + \gamma_k z_k(i-1) + \eta_k x_k(i-1)\right)$ where $x_k(i-1)$ is the first element from the vector $\mathbf{x}_{k,i}$.

Let $\mathcal{X}_i = \{\mathbf{x}_{1,i}; \ldots; \mathbf{x}_{N,i}\}_{\sum_{k=1}^N N_k \times 1}$,

$\mathcal{Y}_i = \{P_1^T \sum_{\ell \in \mathcal{N}_1} P_\ell \mathbf{y}_{\ell,i}; \ldots; P_N^T \sum_{\ell \in \mathcal{N}_N} P_\ell \mathbf{y}_{\ell,i}\}_{\sum_{k=1}^N N_k \times 1}$,

$O_i = \{\mathbf{o}_{1,i}; \ldots; \mathbf{o}_{N,i}\}_{\sum_{k=1}^N N_k \times 1}$,

$\mathcal{Z}_i = \{\mathbf{z}_{1,i}; \ldots; \mathbf{z}_{N,i}\}_{\sum_{k=1}^N N_k \times 1}$ and $A$ be the adjacency matrix of the network with $[A]_{\ell,k} = a_{\ell,k}$ and $\mathcal{A} = A \otimes diag\{I_1; \ldots; I_N\}$, $\mathcal{P} = diag\{P_1; \ldots; P_N\}_{N \times \sum_{k=1}^N N_k}$, and

$$\ddot{\mathcal{G}}_i = \underbrace{\mathcal{A}^T \mathcal{P}^T \mathcal{P} \left( diag\{(I_1 + \ddot{G}_{1,i})^{-1}, \ldots, (I_N + \ddot{G}_{N,i})^{-1}\} \right)}_{\text{Network DLS-LGT Filters.}} \tag{4.75}$$

$r = r_k$, $\mu = \mu_k$, $\alpha' = \alpha'_k$ and $\eta = \eta_k$ for all $k$. Then, we express each node $k$'s recursion from (4.74) for the whole network as

$$\begin{aligned}\mathcal{X}_i &= \frac{1}{(1+r)(1+\eta)} \ddot{\mathcal{G}}_i (\mathcal{Y}_i - O_{i-1} + \gamma \mathcal{Z}_{i-1} + \eta \mathcal{X}_{i-1}) + \frac{r}{1+r} \mathcal{X}_{i-1} \\ &= \frac{1}{(1+r)(1+\eta)} \ddot{\mathcal{G}}_i (\mathcal{Y}_i - O_{i-1} + \gamma \mathcal{Z}_{i-1}) + \Big( \frac{\eta}{(1+r)(1+\eta)} \ddot{\mathcal{G}}_i + \frac{r}{(1+r)} \mathcal{I} \Big) \mathcal{X}_{i-1} \end{aligned} \tag{4.76}$$

Note that $O_i$ and $\mathcal{Z}_i$ are the iterative weighted summation from the rejected components of the local DLS-LGT filters, therefore, we assume $\mathcal{G}_i O_i \approx 0$ and $\mathcal{G}_i \mathcal{Z}_i \approx 0$,

respectively. Thus, Eq. (4.76) becomes

$$\begin{aligned}\mathcal{X}_i &\approx \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i\mathcal{Y}_i + \left(\frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i + \frac{r}{(1+r)}\mathcal{I}\right)\mathcal{X}_{i-1} \\ &= \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i\left(\mathcal{X}_i^o + O_i^o + \boldsymbol{\epsilon}_i\right) + \left(\frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i + \frac{r}{(1+r)}\mathcal{I}\right)\mathcal{X}_{i-1} \qquad (4.77)\end{aligned}$$

For simplicity of analysis, let $\mathbb{E}\ddot{\mathcal{G}}_i = \ddot{\mathcal{G}}_i$ and taking expectation to both side of (4.77), we get

$$\mathbb{E}\mathcal{X}_i = \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i\left(\mathbb{E}\mathcal{X}_i^o + \mathbb{E}O_i^o\right) + \left(\frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i + \frac{r}{(1+r)}\mathcal{I}\right)\mathbb{E}\mathcal{X}_{i-1} \qquad (4.78)$$

Hence, from (4.71), the LGT filter responses are non-negative, therefore, from (4.75), $\rho(\ddot{\mathcal{G}}_i) < 1$ for all $i$. Therefore, we obtain the stability condition of (4.76) as follows;

$$\left\|\frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i + \frac{r}{(1+r)}\mathcal{I}\right\| < 1. \qquad (4.79)$$

With the above condition, $\mathbb{E}\mathcal{X}_i$ converges in the steady state. We assume $\alpha_k' << 1$ for all $k$ and we omit $sgn\{\mathbf{o}_{i-1}\}$ term in (4.72) and write (4.72) for the network as

$$\begin{aligned}O_i &\approx (1-\mu)O_{i-1} + \mu\Big(\mathcal{Y}_i - \frac{1}{(1+\eta)}\mathcal{Y}_i + \frac{1}{1+\eta}O_{i-1} - \frac{\gamma}{1+\eta}\mathcal{Z}_{i-1} - \frac{\eta}{1+\eta}\mathcal{X}_{i-1} + \gamma\mathcal{Z}_{i-1}\Big) \\ &= (1-\mu+\frac{\mu}{1+\eta})O_{i-1} + (\mu - \frac{\mu}{1+\eta})\mathcal{Y}_i + \mu\gamma(1-\frac{1}{1+\eta})\mathcal{Z}_{i-1} - \frac{\mu\eta}{1+\eta}\mathcal{X}_i \\ &= \frac{1+(1-\mu)\eta}{1+\eta}O_{i-1} + \frac{\mu\eta}{1+\eta}\mathcal{Y}_i - \frac{\mu\eta}{1+\eta}\mathcal{X}_{i-1} + \frac{\mu\gamma\eta}{1+\eta}\mathcal{Z}_{i-1} \qquad (4.80)\end{aligned}$$

Now, We take the expected value to the both side of (4.80),

$$\mathbb{E}O_i \approx \frac{1+(1-\mu)\eta}{1+\eta}\mathbb{E}O_{i-1} + \frac{\mu\eta}{1+\eta}\left(\mathbb{E}\mathcal{X}_i^o + \mathbb{E}O_i\right) - \frac{\mu\eta}{1+\eta}\mathbb{E}\mathcal{X}_{i-1} + \frac{\mu\gamma\eta}{1+\eta}\mathbb{E}\mathcal{Z}_{i-1} \qquad (4.81)$$

For the stability of $\mathbb{E}O_i$ in steady state, we need

$$\left|\frac{1+(1-\mu)\eta}{1+\eta}\right| < 1. \qquad (4.82)$$

Therefore, we limit the step size as $0 < \mu < 2$ to achieve the stability of $\mathbb{E}O_i$ in the steady state. Now, from Eq.(4.73), we find $\mathcal{Z}_i$ as

$$\begin{aligned}\mathcal{Z}_i &= \mathcal{Z}_{i-1} + \gamma\left(\mathcal{Y}_i - \frac{1}{(1+\eta)}\mathcal{Y}_i + \frac{1}{1+\eta}O_{i-1} - \frac{\gamma}{1+\eta}\mathcal{Z}_{i-1} - \frac{\eta}{1+\eta}\mathcal{X}_{i-1} - O_i\right)\\ &= \left(1 - \frac{\gamma^2}{1+\eta}\right)\mathcal{Z}_{i-1} + \frac{\gamma\eta}{1+\eta}\mathcal{Y}_i - \gamma O_i + \frac{\gamma}{1+\eta}O_{i-1} - \frac{\gamma\eta}{1+\eta}\mathcal{X}_{i-1}\\ &\approx \left(1 - \frac{\gamma^2}{1+\eta}\right)\mathcal{Z}_{i-1} + \frac{\gamma\eta}{1+\eta}\mathcal{Y}_i - \frac{\gamma\eta}{1+\eta}O_{i-1} - \frac{\gamma\eta}{1+\eta}\mathcal{X}_{i-1} \qquad (4.83)\end{aligned}$$

The expected $\mathcal{Z}_i$ becomes

$$\mathbb{E}\mathcal{Z}_i = \left(1 - \frac{\gamma^2}{1+\eta}\right)\mathbb{E}\mathcal{Z}_{i-1} + \frac{\gamma\eta}{1+\eta}\left(\mathbb{E}\mathcal{X}_i^o + \mathbb{E}O_i^o\right) - \frac{\gamma\eta}{1+\eta}\mathbb{E}O_{i-1} - \frac{\gamma\eta}{1+\eta}\mathbb{E}\mathcal{X}_{i-1}. \qquad (4.84)$$

Hence, for the stability, we limit

$$\begin{aligned}\left|1 - \frac{\gamma^2}{1+\eta}\right| &< 1\\ 0 \le \frac{\gamma^2}{1+\eta} &< 2\\ 0 \le \gamma &< \sqrt{2(1+\eta)} \qquad (4.85)\end{aligned}$$

to ensure $\mathbb{E}\mathcal{Z}_i$ converges in steady state. As in Section 3.5.2, $\lim_{i\to\infty}\mathbb{E}\mathcal{X}_i^o = \bar{\mathcal{X}}^o$ and $\mathbb{E}O_i = \bar{O}^o$ for all $i$ and $\emptyset$ be the all zero matrix of size $\sum_{k=1}^N N_k \times \sum_{k=1}^N N_k$. We stack (4.78), (4.81) and (4.84) such as

$$\begin{bmatrix}\mathbb{E}\mathcal{X}_i\\ \mathbb{E}O_i\\ \mathbb{E}\mathcal{Z}_i\end{bmatrix} = \begin{bmatrix}\frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}} + \frac{r}{1+r}\mathcal{I} & \emptyset & \emptyset\\ -\frac{\mu\eta}{1+\eta}\mathcal{I} & \frac{1+(1-\mu)\eta}{1+\eta}\mathcal{I} & \frac{\mu\eta\gamma}{1+\eta}\mathcal{I}\\ -\frac{\gamma\eta}{1+\eta}\mathcal{I} & -\frac{\gamma\eta}{1+\eta}\mathcal{I} & (1-\frac{\gamma^2}{1+\eta})\mathcal{I}\end{bmatrix}\begin{bmatrix}\mathbb{E}\mathcal{X}_{i-1}\\ \mathbb{E}O_{i-1}\\ \mathbb{E}\mathcal{Z}_{i-1}\end{bmatrix} + \begin{bmatrix}\frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}(\bar{\mathcal{X}}^o + \bar{O}^o)\\ \frac{\mu\eta}{1+\eta}(\bar{\mathcal{X}}^o + \bar{O}^o)\\ \frac{\gamma\eta}{1+\eta}(\bar{\mathcal{X}}^o + \bar{O}^o)\end{bmatrix}. \qquad (4.86)$$

In steady state, we have

$$\begin{bmatrix} \mathbb{E}\mathcal{X}_\infty \\ \mathbb{E}O_\infty \\ \mathbb{E}\mathcal{Z}_\infty \end{bmatrix} = \lim_{i\to\infty} \begin{bmatrix} \mathbb{E}\mathcal{X}_i \\ \mathbb{E}O_i \\ \mathbb{E}\mathcal{Z}_i \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{1+r}\mathcal{I} - \frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}} & \emptyset & \emptyset \\ \frac{\mu\eta}{1+\eta}\mathcal{I} & \frac{\mu\eta}{1+\eta}\mathcal{I} & -\frac{\mu\eta\gamma}{1+\eta}\mathcal{I} \\ \frac{\gamma\eta}{1+\eta}\mathcal{I} & \frac{\gamma\eta}{1+\eta}\mathcal{I} & \frac{\gamma^2}{1+\eta}\mathcal{I} \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}(\bar{\mathcal{X}}^o + \bar{O}^o) \\ \frac{\mu\eta}{1+\eta}(\bar{\mathcal{X}}^o + \bar{O}^o) \\ \frac{\gamma\eta}{1+\eta}(\bar{\mathcal{X}}^o + \bar{O}^o) \end{bmatrix} \tag{4.87}$$

Using the block matrix inversion expression in (4.34), let $A = \frac{1}{1+r}\mathcal{I} - \frac{\eta}{(1+\eta)}\ddot{\mathcal{G}}$, $B = \{\emptyset \quad \emptyset\}$, $C = \{\frac{\mu\eta}{1+\eta}\mathcal{I}; \frac{\gamma\eta}{1+\eta}\mathcal{I}\}$,

$$D = \begin{bmatrix} \frac{\mu\eta}{1+\eta}\mathcal{I} & -\frac{\mu\eta\gamma}{1+\eta}\mathcal{I} \\ \frac{\gamma\eta}{1+\eta}\mathcal{I} & \frac{\gamma^2}{1+\eta}\mathcal{I} \end{bmatrix}; \quad D^{-1} = \begin{bmatrix} \frac{1}{\mu\eta}\mathcal{I} & \frac{1}{\gamma}\mathcal{I} \\ -\frac{1}{\mu\gamma}\mathcal{I} & \frac{1}{\gamma^2}\mathcal{I} \end{bmatrix} \tag{4.88}$$

and we arrive the expected value of (4.87) in steady state as,

$$\begin{bmatrix} \mathbb{E}\mathcal{X}_\infty \\ \mathbb{E}O_\infty \\ \mathbb{E}\mathcal{Z}_\infty \end{bmatrix} = \begin{bmatrix} \left(\frac{1}{1+r}\mathcal{I} - \frac{\eta}{(1+r)(1+\eta)}\mathcal{G}\right)^{-1} & \emptyset & \emptyset \\ -\left(\frac{1}{1+r}\mathcal{I} - \frac{\eta}{(1+r)(1+\eta)}\mathcal{G}\right)^{-1} & \frac{1}{\mu\eta}\mathcal{I} & \frac{1}{\gamma}\mathcal{I} \\ \emptyset & -\frac{1}{\mu\gamma}\mathcal{I} & \frac{1}{\gamma^2}\mathcal{I} \end{bmatrix} \begin{bmatrix} \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}(\bar{\mathcal{X}}^o + \bar{O}^o) \\ \frac{\mu\eta}{1+\eta}\left(\bar{\mathcal{X}}^o + \bar{O}^o\right) \\ \frac{\gamma\eta}{1+\eta}\left(\bar{\mathcal{X}}^o + \bar{O}^o\right) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{(1+\eta)}\left(\mathcal{I} - \frac{\eta}{(1+\eta)}\ddot{\mathcal{G}}\right)^{-1}\ddot{\mathcal{G}}\left(\bar{\mathcal{X}}^o + \bar{O}^o\right) \\ \left(\mathcal{I} - \frac{\eta}{(1+\eta)}\ddot{\mathcal{G}}\right)^{-1}\left(\mathcal{I} - \ddot{\mathcal{G}}\right)\left(\bar{\mathcal{X}}^o + \bar{O}^o\right) \\ \mathbf{0} \end{bmatrix} \tag{4.89}$$

where $\mathbf{0}$ is an all-zero vector. Notice that in (4.89), $\mathbb{E}\mathcal{X}_\infty$ and $\mathbb{E}O_\infty$ do not converge to $\bar{\mathcal{X}}^o$ and $\bar{O}^o$, respectively. Therefore, we re-write $\mathbb{E}\mathcal{X}_\infty$ and $\mathbb{E}O_\infty$ from (4.89) as;

$$\mathbb{E}\mathcal{X}_\infty = \bar{\mathcal{X}}^o - \underbrace{\left(\mathcal{I} - \frac{\eta}{1+\eta}\mathcal{G}\right)^{-1}(\mathcal{I} - \mathcal{G})\,\bar{\mathcal{X}}^o}_{\substack{\text{True signal rejected by} \\ \text{LGT filters.}}} + \underbrace{\frac{1}{1+\eta}\left(\mathcal{I} - \frac{\eta}{1+\eta}\mathcal{G}\right)^{-1}\ddot{\mathcal{G}}\bar{O}^o}_{\substack{\text{Outliers accepted by} \\ \text{LGT filters.}}} \tag{4.90}$$

$$\mathbb{E}O_{\infty} = \bar{O}^{o} - \underbrace{\frac{1}{(1+\eta)}\left(\mathcal{I} - \frac{\eta}{1+\eta}\mathcal{G}\right)^{-1}\ddot{\mathcal{G}}\bar{O}^{o}}_{\substack{\text{Outliers accepted by} \\ \text{LGT filters.}}} + \underbrace{\left(\mathcal{I} - \frac{\eta}{(1+\eta)}\ddot{\mathcal{G}}\right)^{-1}\left(\mathcal{I} - \ddot{\mathcal{G}}\right)\bar{\mathcal{X}}^{o}}_{\substack{\text{True signal rejected from} \\ \text{LGT filters.}}}. \tag{4.91}$$

Notice that $\mathbb{E}\mathcal{X}_{\infty}$ converges to the sum of the true signal and outliers filtered by the LS-LGT filters and notice that $\mathbb{E}O_{\infty}$ converges to the sum of the true outliers and the other two extra terms; the outliers filtered by the DLS-LGT filters and the rejected component of the true signal from the filters. Hence the estimators have bias. However, with the careful selection of DLS-LGT filters which can filter out the outliers from the true signal, then we can achieve asymptotically unbiased estimator, i.e.,

$$\frac{1}{1+\eta}\left(\mathcal{I} - \frac{\eta}{1+\eta}\mathcal{G}\right)^{-1}\ddot{\mathcal{G}}\bar{\mathcal{X}}^{o} \approx \bar{\mathcal{X}}^{o}; \quad \ddot{\mathcal{G}}\bar{O}^{o} \approx 0 \tag{4.92}$$

and

$$\mathbb{E}\mathcal{X}_{\infty} \approx \bar{\mathcal{X}}^{o}; \quad \mathbb{E}O_{\infty} \approx \bar{O}^{o}. \tag{4.93}$$

### 4.6.2 Error Recursion and Bias

Now, to evaluate the performance of the proposed LS-LGT method, we introduce the two error vector; $\tilde{\mathcal{X}}_i$ and $\tilde{O}_i$ defined as the network learning errors of $\mathcal{X}_i^{o}$ and $\bar{O}^{o}$

at time $i$, respectively, defined as

$$\begin{aligned}
\begin{bmatrix} \tilde{\mathcal{X}}_i \\ \tilde{O}_i \end{bmatrix} &= \begin{bmatrix} \mathcal{X}_i^o \\ O^o \end{bmatrix} - \begin{bmatrix} \mathcal{X}_i \\ O_i \end{bmatrix} \\
&= \begin{bmatrix} \left(\frac{r}{1+r}\mathcal{I} + \frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right) & \emptyset \\ -\frac{\mu\eta}{1+\eta}\mathcal{I} & \frac{1+(1-\mu)\eta}{1+\eta}\mathcal{I} \end{bmatrix} \begin{bmatrix} \tilde{\mathcal{X}}_{i-1} \\ \tilde{O}_{i-1} \end{bmatrix} \\
&\quad + \begin{bmatrix} \left(\frac{a-r}{a(1+r)}\mathcal{I} - \frac{(a+\eta)}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}_i\right)\mathcal{X}_i^o \\ \frac{(1-a)\mu\eta}{a(1+\eta)}\mathcal{X}_i^o \end{bmatrix} + \begin{bmatrix} \left(\frac{r}{a(1+r)}\mathcal{I} + \frac{\eta}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}_i\right)Q_i \\ -\frac{\mu\eta}{a(1+\eta)}Q_i \end{bmatrix} \\
&\quad - \begin{bmatrix} \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i O_i^o \\ \emptyset \end{bmatrix} - \begin{bmatrix} \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}_i\boldsymbol{\epsilon}_i \\ \frac{\mu\eta}{(1+\eta)}\boldsymbol{\epsilon}_i. \end{bmatrix}
\end{aligned} \tag{4.94}$$

Taking expectation to both side of (4.94),

$$\begin{aligned}
\begin{bmatrix} \mathbb{E}\tilde{\mathcal{X}}_i \\ \mathbb{E}\tilde{O}_i \end{bmatrix} &= \begin{bmatrix} \left(\frac{r}{1+r}\mathcal{I} + \frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right) & \emptyset \\ \frac{-\mu\eta}{1+\eta}\mathcal{I} & \frac{1+(1-\mu)\eta}{1+\eta}\mathcal{I} \end{bmatrix} \begin{bmatrix} \mathbb{E}\tilde{\mathcal{X}}_{i-1} \\ \mathbb{E}\tilde{O}_{i-1} \end{bmatrix} + \begin{bmatrix} \left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{(a+\eta)}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathbb{E}\mathcal{X}_i^o \\ \frac{(1-a)\mu\eta}{a(1+\eta)}\mathbb{E}\mathcal{X}_i^o \end{bmatrix} \\
&\quad + \begin{bmatrix} \left(\frac{r}{a(1+r)}\mathcal{I} + \frac{\eta}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathbb{E}Q_i \\ \frac{-\mu\eta}{a(1+\eta)}\mathbb{E}Q_i \end{bmatrix} + \begin{bmatrix} \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\bar{O}^o \\ \emptyset \end{bmatrix}.
\end{aligned} \tag{4.95}$$

Since $\lim_{i\to\infty}\mathbb{E}\mathcal{X}_i^o = \bar{\mathcal{X}}^o$, $\bar{Q} = (1-a)\bar{\mathcal{X}}^o$, we get the steady state expected network's learning errors as

$$\begin{aligned}
\begin{bmatrix} \mathbb{E}\tilde{\mathcal{X}}_\infty \\ \mathbb{E}\tilde{O}_\infty \end{bmatrix} &= \lim_{i\to\infty} \begin{bmatrix} \mathbb{E}\tilde{\mathcal{X}}_i \\ \mathbb{E}\tilde{O}_i \end{bmatrix} \\
&= \begin{bmatrix} \left(\frac{r}{1+r}\mathcal{I} - \frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right) & \emptyset \\ \frac{\mu\eta}{1+\eta}\mathcal{I} & \frac{\mu\eta}{1+\eta}\mathcal{I} \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{1+r}\left(\mathcal{I} - \ddot{\mathcal{G}}\right)\bar{\mathcal{X}}^o \\ \emptyset \end{bmatrix} \\
&\quad - \begin{bmatrix} \left(\frac{r}{1+r}\mathcal{I} - \frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right) & \emptyset \\ \frac{\mu\eta}{1+\eta}\mathcal{I} & \frac{\mu\eta}{1+\eta}\mathcal{I} \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\bar{O}^o \\ \emptyset \end{bmatrix}.
\end{aligned} \tag{4.96}$$

Using the expression in (4.34), we write the matrix inverse in (4.96) as

$$\begin{bmatrix} \left(\frac{1}{1+r}\mathcal{I} - \frac{\eta}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right) & \emptyset \\ \frac{\mu\eta}{1+\eta}\mathcal{I} & \frac{\mu\eta}{1+\eta}\mathcal{I} \end{bmatrix}^{-1} = \begin{bmatrix} \left(\frac{1}{1+r}\mathcal{I} - \frac{\eta}{\alpha(1+\eta)}\ddot{\mathcal{G}}\right)^{-1} & \emptyset \\ -\left(\frac{1}{1+r}\mathcal{I} - \frac{\eta}{\alpha(1+\eta)}\ddot{\mathcal{G}}\right)^{-1} & \frac{1+\eta}{\mu\eta}\mathcal{I} \end{bmatrix}. \tag{4.97}$$

Then, we have the expected errors in steady state as,

$$\mathbb{E}\tilde{\mathcal{X}}_\infty = \underbrace{\left(\mathcal{I} - \frac{\eta}{(1+\eta)}\ddot{\mathcal{G}}\right)^{-1}\left(I - \ddot{\mathcal{G}}\right)\bar{\mathcal{X}}^o}_{\substack{\text{True signal rejected by} \\ \text{LGT filters.}}} - \underbrace{\left((1+\eta)\mathcal{I} - \eta\ddot{\mathcal{G}}\right)^{-1}\ddot{\mathcal{G}}\bar{O}^o}_{\substack{\text{Outliers accepted by} \\ \text{LGT filters.}}} < \infty;$$

$$\mathbb{E}\tilde{\mathcal{X}}_\infty = -\mathbb{E}\tilde{O}_\infty. \tag{4.98}$$

### 4.6.3 Mean-Square Stability

From (4.98), we have the learning errors from (4.94) converge in steady state. Now, we will study the mean-square behavior of these network errors. Let $\ddot{\mathcal{B}}_i = \left(\frac{r}{1+r}\mathcal{I} + \frac{\eta}{(1+r)(1+\eta)}\ddot{\ddot{\mathcal{G}}}_i\right)$ and $\mathbb{E}\ddot{\mathcal{B}}_i = \ddot{\mathcal{B}}$. From (4.94), we find the mean-squared error for estimating $\mathcal{X}_i^o$ as

$$\begin{aligned}
\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 = {} & \mathbb{E}\|\ddot{\mathcal{B}}_i\tilde{\mathcal{X}}_{i-1}\|^2 + \frac{1}{(1+r)^2(1+\eta)^2}\ddot{\mathcal{G}}\mathbb{E}\left(\boldsymbol{\epsilon}_i\boldsymbol{\epsilon}_i^T\right)\ddot{\mathcal{G}}^T + \frac{1}{(1+r)^1(1+\eta)^2}\ddot{\mathcal{G}}\bar{O}^o\bar{O}^o\ddot{\mathcal{G}}^T \\
& + \left(\frac{r}{a(1+r)}\mathcal{I} + \frac{\eta}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathbb{E}\left(Q_iQ_i^T\right)\left(\frac{r}{a(1+r)}\mathcal{I} + \frac{\eta}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)^T \\
& + \left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{(a+\eta)}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathbb{E}\left(\mathcal{X}_i^o\mathcal{X}_i^{oT}\right) \\
& \cdot\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{(a+\eta)}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)^T \\
& + 2\ddot{\mathcal{B}}\mathbb{E}\tilde{\mathcal{X}}_i\left(\left(\frac{a(1+r)-r}{a(1+r)}\mathcal{I} - \frac{a+\eta}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathbb{E}\mathcal{X}_i^o\right. \\
& \left. + \left(\frac{r}{a(1+r)}\mathcal{I} + \frac{\eta}{a(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathbb{E}Q_i\right)^T
\end{aligned} \tag{4.99}$$

Using (4.95), we express the mean-squared error for $\tilde{\mathcal{X}}_i$ as

$$\begin{aligned}\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 &= \lim_{i\to\infty}\mathbb{E}\|\tilde{\mathcal{X}}_i\|^2 \\ &\approx \underbrace{\|\mathbb{E}\tilde{\mathcal{X}}_\infty\|^2}_{\text{bias squared.}} + \underbrace{\sum_{j=0}^{\infty}\ddot{\mathcal{B}}^j\left(I-\frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)\mathcal{P}^T L^\dagger\mathcal{P}\left(I-\frac{1}{(1+r)(1+\eta)}\ddot{\mathcal{G}}\right)^T\ddot{\mathcal{B}}^{jT}}_{\text{error due to non-stationary signal model.}} \\ &\quad + \underbrace{\frac{1}{(1+r)^2(1+\eta)^2}tr\left(\sum_{j=0}^{\infty}\ddot{\mathcal{B}}^j\ddot{\mathcal{G}}\mathcal{P}^T\Sigma\mathcal{P}\ddot{\mathcal{G}}^T\ddot{\mathcal{B}}^{jT}\right)}_{\text{error due to measurement noise.}} \end{aligned} \tag{4.100}$$

Similarly, we find the mean-squared error for estimating the outlier $O_i^o$ as

$$\begin{aligned}\mathbb{E}\|\tilde{O}_i\|^2 &= \frac{\left(1+(1-\mu)\eta\right)^2}{(1+\eta)^2}\mathbb{E}\|\tilde{O}_{i-1}\|^2 + \frac{\mu^2\eta^2}{(1+\eta)^2}\mathbb{E}\|\boldsymbol{\epsilon}_i\|^2 + \frac{\mu^2\eta^2}{(1+\eta)^2}\mathbb{E}\|\tilde{\mathcal{X}}_{i-1}\|^2 \\ &\quad + \frac{(1-a)^2\mu^2\eta^2}{a^2(1+\eta)^2}\mathbb{E}\|\mathcal{X}_i^o\|^2 + \frac{\mu^2\eta^2}{a^2(1+\eta)^2}\mathbb{E}\|Q_i\|^2 \\ &\quad + 2\frac{(1+(1-\mu)\eta)\mu\eta}{(1+\eta)^2}\mathbb{E}\left(\tilde{O}_{i-1}\tilde{\mathcal{X}}_{i-1}^T\right) + \frac{2(1-a)\mu\eta}{a(1+\eta)}\mathbb{E}\left(\mathcal{X}_i^o Q_i^T\right) \\ &\quad + 2\left(\frac{(1+(1-\mu)\eta)}{(1+\eta)}\mathbb{E}\tilde{O}_{i-1} + \frac{\mu\eta}{(1+\eta)}\mathbb{E}\tilde{\mathcal{X}}_{i-1}\right)\left(\frac{(1-a)\mu\eta}{a(1+\eta)}\mathbb{E}\mathcal{X}_i^o - \frac{\mu\eta}{a(1+\eta)}\mathbb{E}Q_i\right)^T. \end{aligned} \tag{4.101}$$

With (4.95), we can express (4.101) to a more compact form as

$$\begin{aligned}\mathbb{E}\|\tilde{O}_i\|^2 &\approx \frac{(1+(1-\mu)\eta)^2}{(1+\eta)^2}\mathbb{E}\|\tilde{O}_{i-1}\|^2 + \frac{\mu^2\eta^2}{(1+\eta)^2}\mathbb{E}\|\boldsymbol{\epsilon}_i\|^2 + \frac{\mu^2\eta^2}{(1+\eta)^2}\mathbb{E}\|Q_i\|^2 \\ &\quad + \|\mathbb{E}\tilde{O}_i\|^2 - \frac{(1+(1-\mu)\eta)^2}{(1+\eta)^2}\|\mathbb{E}\tilde{O}_{i-1}\|^2 + \frac{\mu^2\eta^2}{(1+\eta)^2}\left(\mathbb{E}\|\tilde{\mathcal{X}}_{i-1}\|^2 - \|\mathbb{E}\tilde{\mathcal{X}}_{i-1}\|^2\right) \end{aligned} \tag{4.102}$$

Hence, in steady state, we conclude that

$$\begin{aligned}\mathbb{E}\|\tilde{O}_\infty\|^2 &= \lim_{i\to\infty}\mathbb{E}\|\tilde{O}_i\|^2 \\ &\approx \|\mathbb{E}\tilde{O}_\infty\|^2 + tr\Big(\frac{\mu\eta}{2+2\eta-\mu\eta}\Sigma + \frac{\mu\eta}{2+2\eta-\mu\eta}L^\dagger\Big) \\ &\quad + \frac{\mu\eta}{2+2\eta-\mu\eta}(\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 - \|\mathbb{E}\tilde{\mathcal{X}}_\infty\|^2). \end{aligned} \tag{4.103}$$

Now, let $MSD_x^r$ and $MSD_o^r$ be the network mean square deviation for estimating $\mathcal{X}_i^o$ and $O_i^o$ in steady state, defined as,

$$MSD_x^r = \frac{1}{\sum_{k=0}^{N} N_k}\mathbb{E}\|\tilde{\mathcal{X}}_\infty\|^2 < \infty; \quad MSD_o^r = \frac{1}{\sum_{k=0}^{N} N_k}\mathbb{E}\|\tilde{O}_\infty\|^2 < \infty. \tag{4.104}$$

## 4.7 Experiments

### 4.7.1 Experiments on Synthetic Data

In this section, we simulate the estimation and tracking performance of the graph signal and outlier using the proposed DLS-LGT outliers method. We compare the proposed LGT based methods with the transitional distributed Least Square solution and outliers estimation by setting the LGT filter coefficients to zero in Alg.7 and we denote as dist.LS-outliers. We selected 200 nodes randomly distributed over $2D$ plane and formed the graph based on the Euclidean distance between the nodes. In order the effectively evaluate the tracking performance of the algorithm we use the non-statinary data model mentioned in (2.4) with $\mathbf{q}_i \sim \mathcal{N}(\mathbf{0}, 0.1L^\dagger)$. We injected the burst outliers to the ground truth signal and $SNR = 0dB$. Figure 4.17 shows the MSD comparison of estimating the graph signal and the outliers between the transitional distributed LS and the proposed DLS-LGT-outliers algorithm for different $r$ values. The tracking performance are shown in Figure 4.16. Figure 4.15 have the ROC comparison of the detection results from the window GLRT on the estimated outliers for different $r$ and the window size of 30.

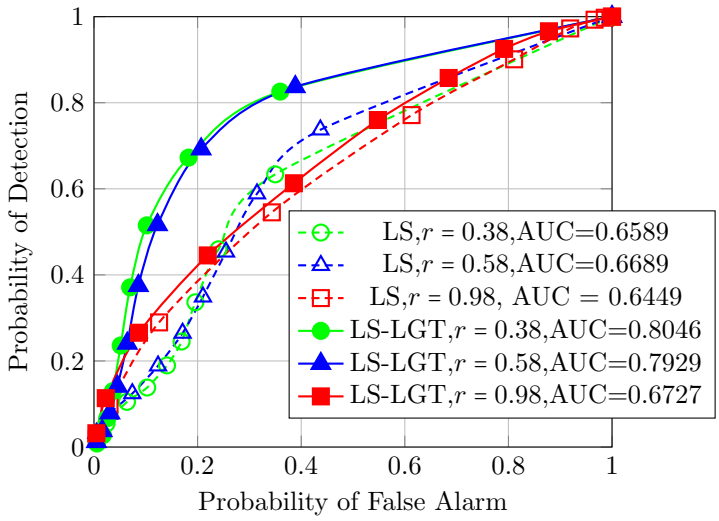



Figure 4.15: ROC comparison of the detection of the burst outliers from the trandition dist. LS-outliers and the proposed DLS-LGT-ouliters Alg.7 for different value of $r$.

### 4.7.2 Experiments on Real-World Dataset

Similar in Section 4.4.2, we evaluated the performance of DLS-LGT-ouliters Alg.6 on the weather dataset, [65], the Abilene Backbone dataset, [66] and the UGR'16 traffic dataset, [67], respectively. The experiment setting are mentioned in Section 4.4.2. We compare the tracking performance of the DLS-LGT oultiers algorithm for $r = 0.18, 0.58$ and $0.98$. We corrupted the true signal with the burst outliers as mentioned in Section 4.4.2. Then, we generate the ROC comparison by using the window GLRT on the estimated outliers signal from the DLS-LGT-outliers and the traditional dist.LS-outliers algorithm without the LGT fliters.

Figure 4.18 and 4.21 show the ROC comparison and the tracking performance of the DLS-LGT-outliers Alg.7 on the weather dataset, [65].

The results on the Abilene backbone network, [66] are given in Figure 4.19 and 4.22. We have the ROC comparison for the detection of the burst outliers between the

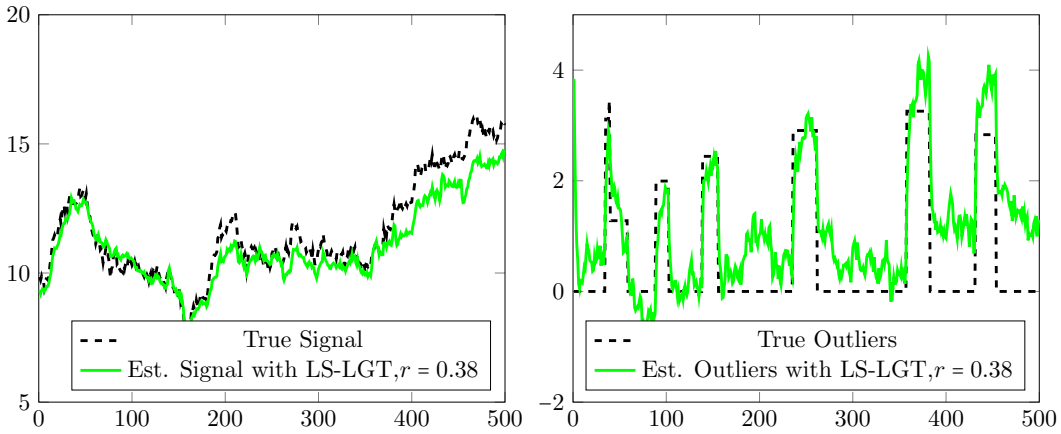


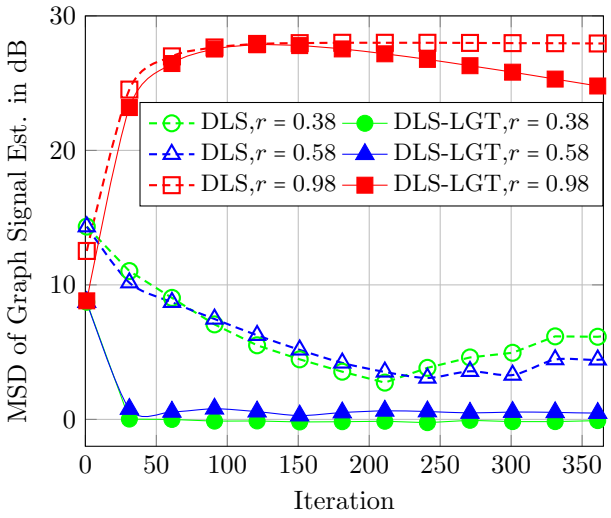

Figure 4.16: Tracking performance of the DLS-LGT-outliers Alg.7.

dist.LS-outliers and the DLS-LGT-outliers Alg.7 in Figure 4.19. Figure 4.22 shows the tracking performance of the estimated traffic and outliers by using DLS-LGT-outliers for $r = 0.18, 0.58$ and $0.98$, respectively.

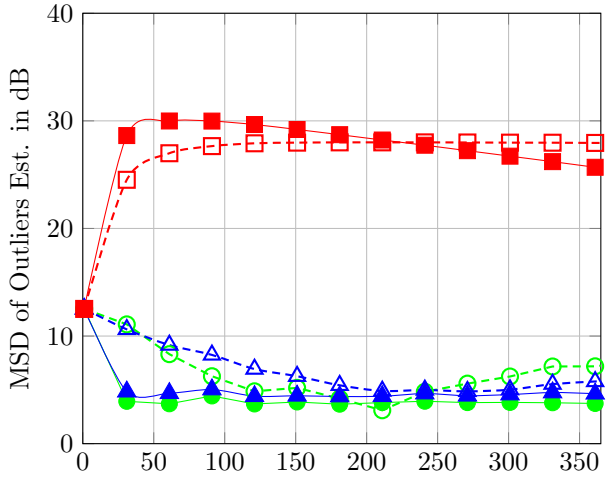
Figure 4.23 and 4.24 show the tracking performance of the estimated benign traffic and outliers in UGR'16 traffic, [67], respectively. The ROC comparison between the two algorithms for different $r$ values is in Figure 4.20. As we can see from the results, we improves the performance of the signal and outlier estimation with the proposed DLS-LGT-outliers Alg.7 with the LGT filters in comparison to the traditional algorithms.

## 4.8 Conclusion

We proposed the distributed solutions based on adaptive filtering and optimization principles for the tracking of both a non-stationary signal emanating from a time-varying graph, and sparse outliers , both corrupted by noise. The LGT filtering ensure local signal estimates are smooth with respect to the local graph, with additional




(a)




(b)

Figure 4.17: MSD Comparison of the graph signal and ouliters estimation between the tradition DLS-outliers and the proposed DLS-LGT-outliers Alg.7. $SNR = 0dB$.

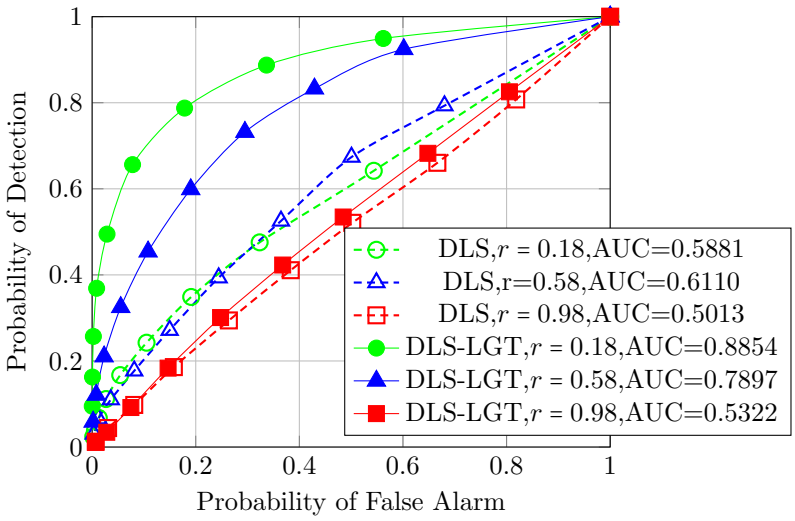



Figure 4.18: ROC comparison of the detection of the burst outliers from the dist.LS-outliers and the DLS-LGT-outliers Alg.7 for different $r$ value on the Weather Data, [65].

benefits for outlier estimation. With less computation complexity, the LGT method are robust and suitable for time varying graphs.

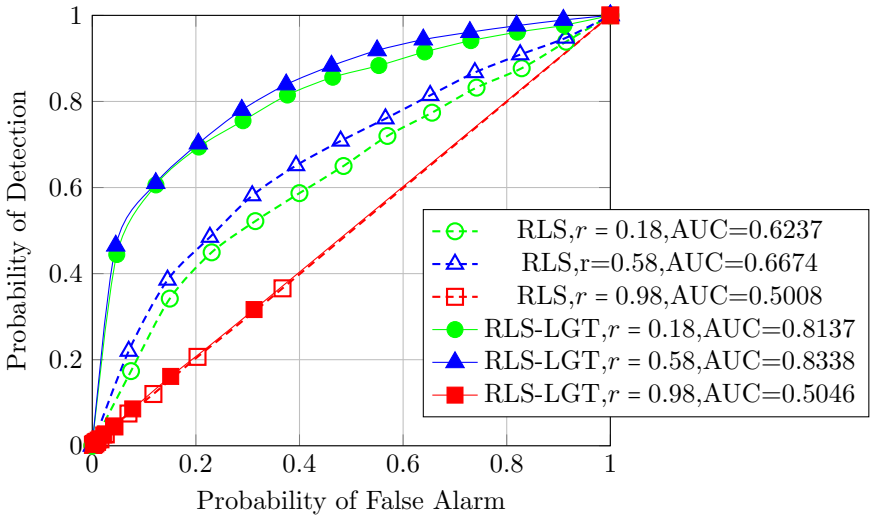



Figure 4.19: ROC comparison of detecting the burst outliers between the dist. LS-outliers and DLS-LGT-outliers Alg.7 for Abilene Backbone Network, [66].

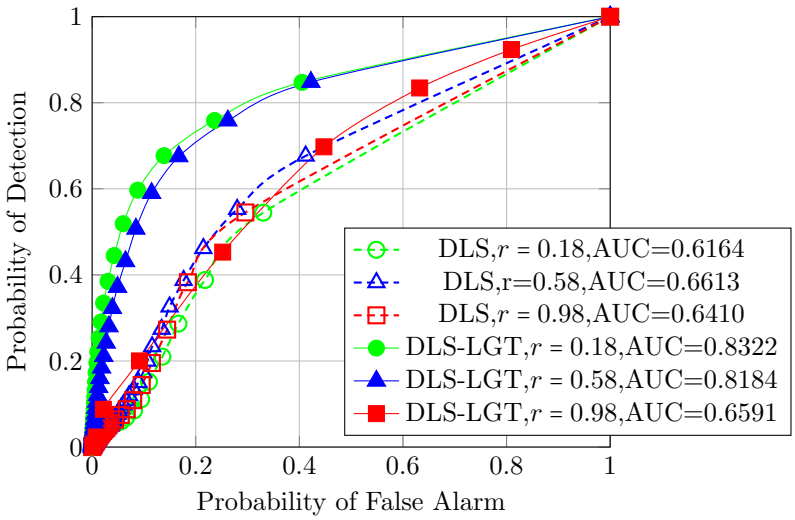



Figure 4.20: ROC comparison of detecting the burst outliers between the dist. LS-outliers and DLS-LGT-outliers Alg.6 for UGR'16 dataset, [67].

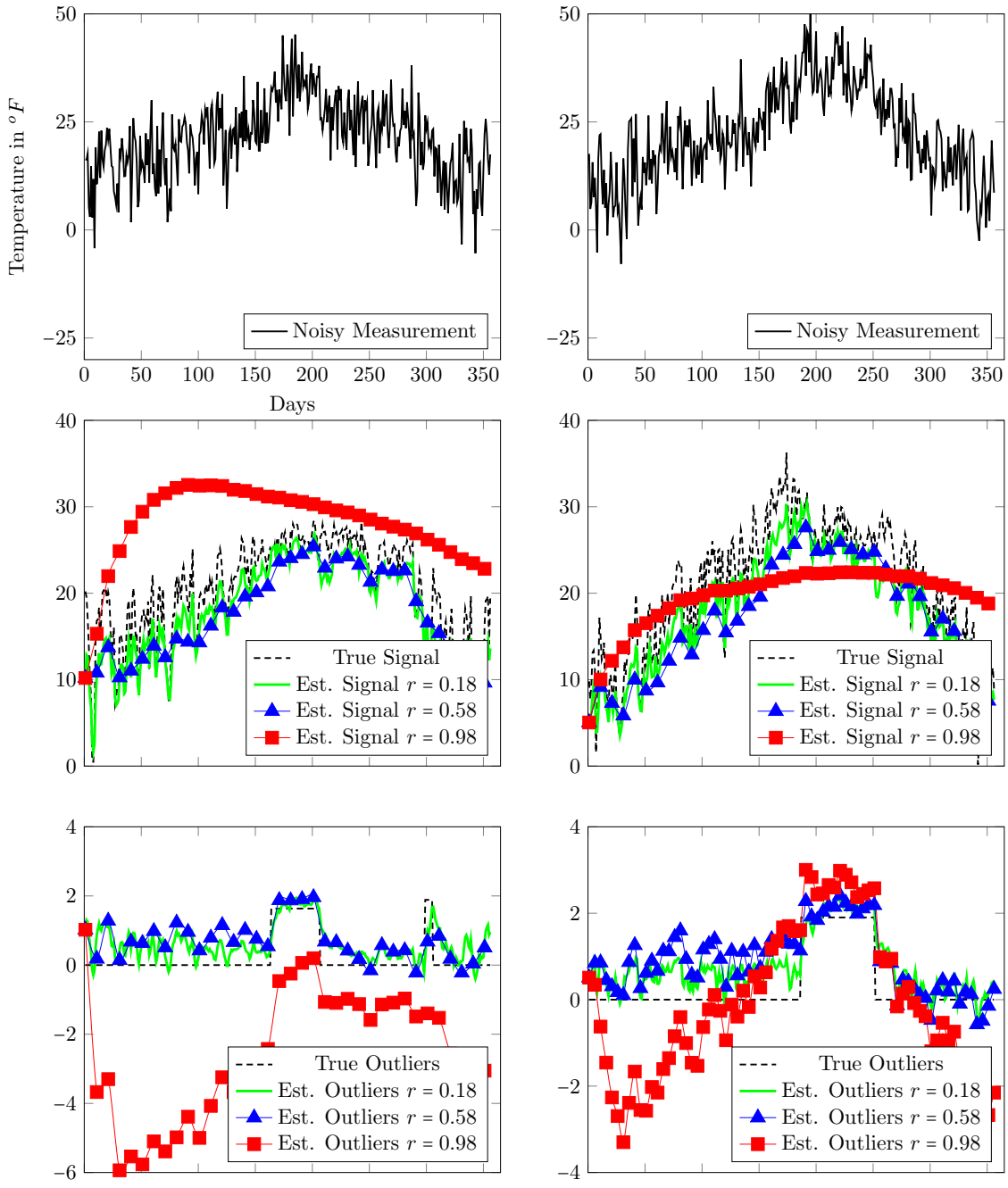



Figure 4.21: Tracking performance of the true temperature and outlier estimation from DLS-LGT-outliers Alg.7 on Weather Dataset, [65].

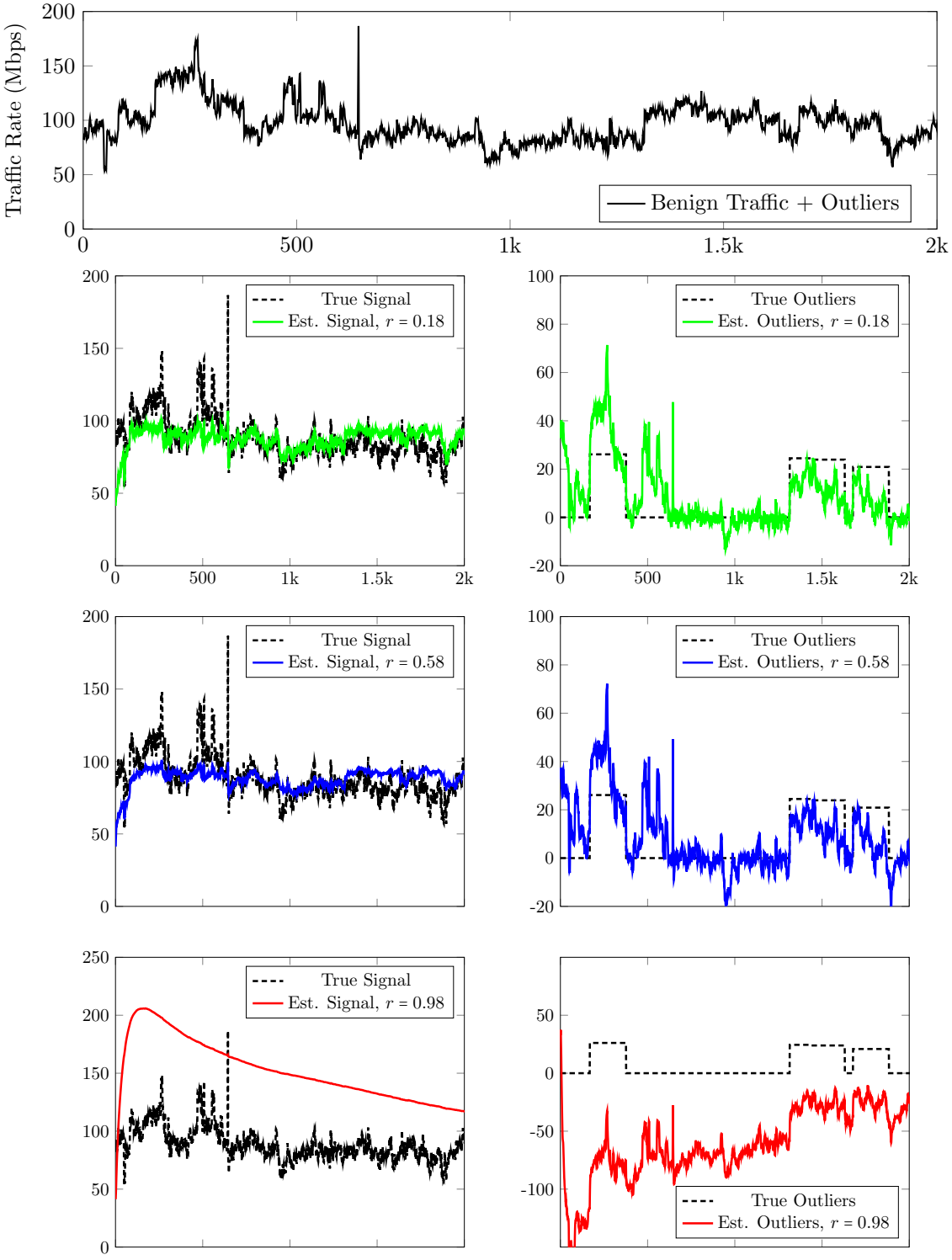



Figure 4.22: Tracking Performance of LS-LGT on Abilene Backbone Network.

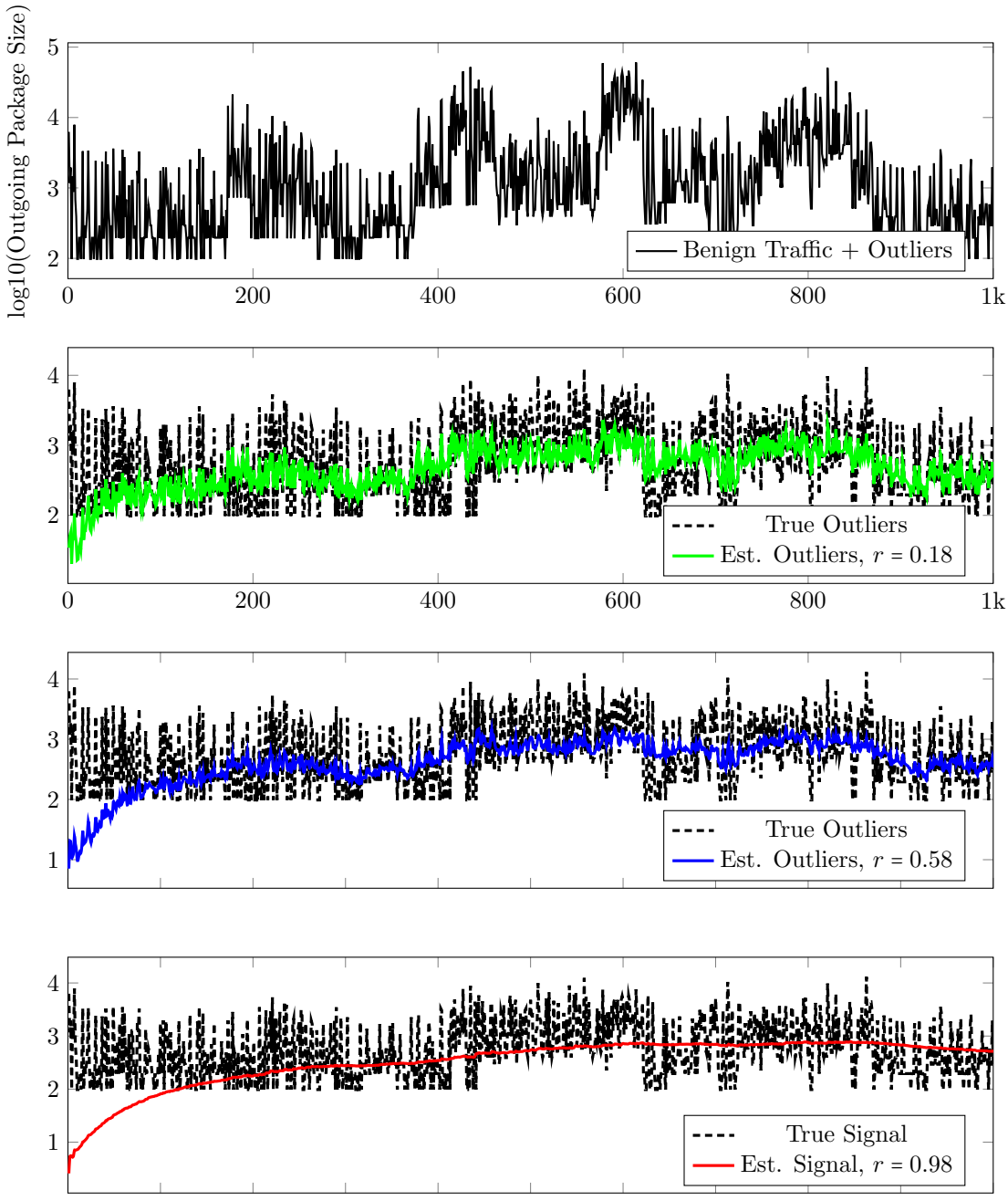



Figure 4.23: Benign traffic tracking performance of DLS-LGT oultiers on UGR'16 Dataset, [67].

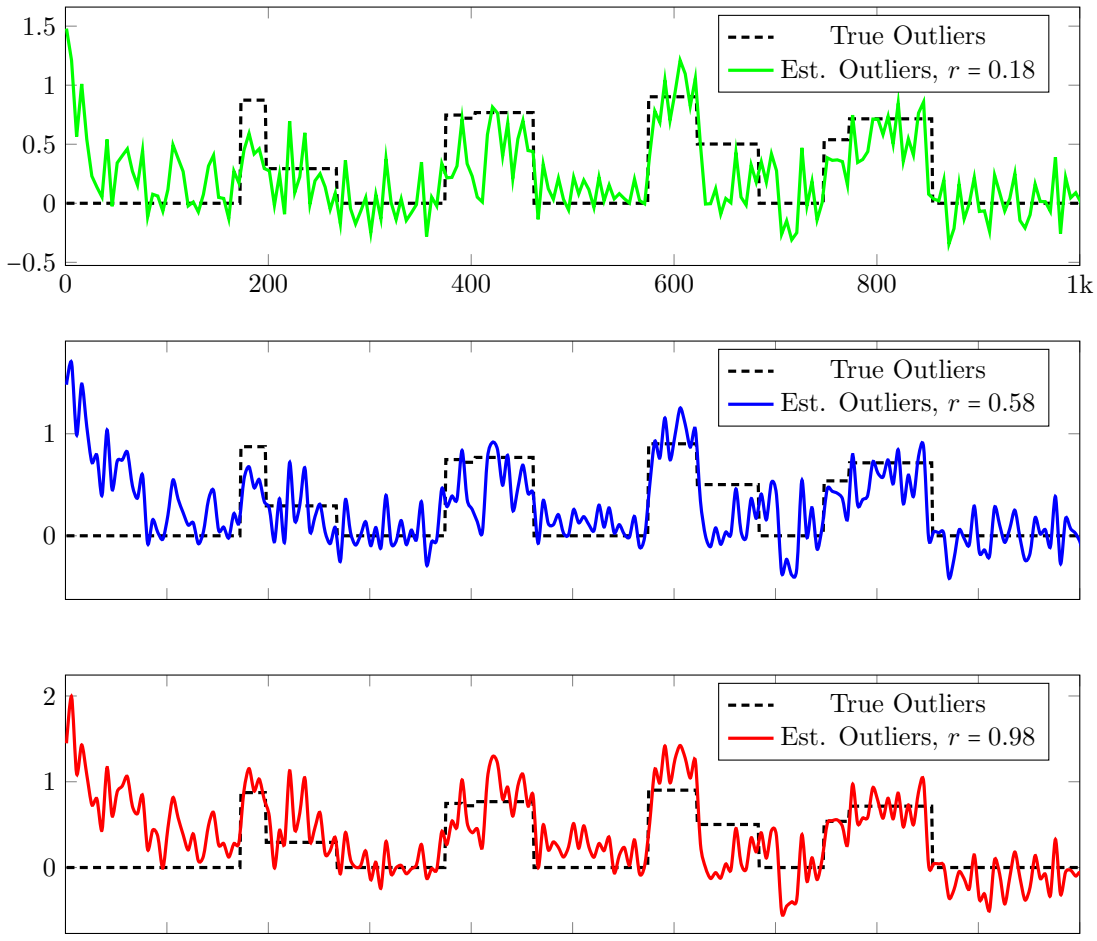



Figure 4.24: Outliers tracking Performance of DLS-LGT-Outliers on UGR'16 Dataset,. [67]. The corrupted benign traffic is shown in Figure 4.23.

# CHAPTER 5

# Conclusion and Future Work

## 5.1 Conclusion

This research examines the problem of the motivation for estimating and tracking non-stationary signals residing on static/time-varying graphs in the presence of the anomaly /outlier signals. We presented the optimal graph filter design to achieve performance gain over the existing methods. We proposed the distributed scheme for tracking the graph signal and introduce the Local Graph Transform which does not rely on global/centralized information. The LGT-based methods are scalable to large graphs with growing numbers of nodes, and edge weight changes. The analytical and simulation results show the LGT-based estimation achieve the performance gain over the traditional diffusion methods.

## 5.2 Future Work

Beyond the research work presented in this thesis, some of the future directions of the research can be listed as

1. Local Graph Transform for directed-graphs

2. Efficient updating of LGT for time-varying graphs

3. Learning and updating the graphical structure of non-stationary signals

4. Incorporating LGT in other adaptive algorithms

5. LGT and geometric deep learning.

**Local Graph Transform for directed-graph**

There are two main approaches for processing a graph signal using GFT: the Laplacian matrix based approach and weighted adjacency matrix approaches. The former approach mostly considers the signal lying on an undirected graph structure and performs the GFT frequency analysis with the eigendecomposition of the positive semi-definite Laplacian matrix [18]. Therefore, this approach is limited to unidirected graphs. For directed graphs, one can use the symmetrized Laplacian of the directed graph. However, it cannot capture the edges' directivity [72]. When using the weighted adjacency as a GFT operator, the Jordan Canonical form can be used as the GFT matrix to transform the graph signal into GFT domain [48]. Although one can use this approach to process signals on a directed graph [73], the Jordan decomposition becomes computationally unstable even for a moderate size matrix [74]. In [72], the author proposed an approach to build an orthogonal GFT basis for directed graphs. While these approaches are based on the global information, we are interested in local processing which does not require any global information. Can we form a Local Directed-Graph Transform and incorporate it as a local penalty function to minimize the total variation in the neighborhood? As one of our subjects

for future work, one can further explore and analyze our proposed LGT methods on the directed graph structure.

**Efficient updating of LGT for time-varying graphs**

Another direction of this research is to explore the efficient updating of the LGT for time varying graphs. Although the proposed LGT method reduces the computational complexity in comparison to the centralized GFT method, in the worst scenario, updating all the eigenvalues and eigenvectors of LGT requires the computation complexity of $\mathcal{O}(N.\bar{N}_k^3)$. The computational cost of LGT can become quite expensive if the average number of neighbors in the network becomes large, especially for time varying graphs. However, as all the local Laplacians in the network have a star topology, one can explore an efficient method for updating the eigendecomposition of the time-varying local Laplacian in order to reduce the complexity.

**Learning and updating the graphical structure of non-stationary signals**

One of the fundamental challenges in the GSP is learning the proper graph representation of the signal. In many GSP applications, researchers use the graphs which are known a prior (e.g., social networks) or formed from the prior knowledge of signal domain (e.g., geodesic distance). However, these graphs may not truly represent the underlying structure of the signal. Although much research has been proposed in finding the smooth graph representation of the signal [62, 68, 69, 71, 75], it is not clear how to efficiently update such graph structures for non-stationary signals. It can be computationally challenging to learn the time-varying graphs frequently from the non-stationary signals. Therefore, another interesting future direction of our research

would be exploring an efficient method to learn the underlying local graph (LGT) structure of the non-stationary signal.

**Incorporating LGT in other adaptive algorithms**

In Chapter 3 and 4, we studied the adaptive algorithms: diffusion LMS and RLS with the LGT-filters for graph signal and outliers estimation. However, there are many different variations of the distributed adaptive algorithms proposed for different applications such as diffusion normalized LMS for fast convergence with low steady-state error [76], diffusion leaky LMS in echo cancellation [77] and diffusion sign-error LMS for impulsive interference [78], etc. Therefore, one can compare the performance of these variant adaptive algorithms with the LGT filters. In addition, one can explore approaches that reduce the amount of message passing, or rely on asynchronous updates, which might be required in certain scenarios.

**LGT and geometric deep learning**

In recent years, GSP has become a promising tool in statistical signal processing and many researchers have integrated GSP tools into deep learning [79–82]. In particular, the GFT and graph filters are utilized in neural network architectures. This GFT-utilization leads to better performance, [80]. However, these methods are focusing on learning and training on a centralized GFT, one can try exploring on the effectiveness of neutral networks using the proposed LGT-filters as a continuation of this research.

# Bibliography


[1] A. H. Sayed, *Adaptive filters*. John Wiley & Sons, 2011.

[2] ——, "Diffusion adaptation over networks," *CoRR*, vol. abs/1205.4220, 2012. [Online]. Available: http://arxiv.org/abs/1205.4220

[3] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4289–4305, Aug 2012.

[4] X. Zhao, S. Y. Tu, and A. H. Sayed, "Diffusion adaptation over networks under imperfect information exchange and non-stationary data," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3460–3475, July 2012.

[5] A. H. Sayed and C. G. Lopes, "Distributed recursive least-squares strategies over adaptive networks," in *2006 Fortieth Asilomar Conference on Signals, Systems and Computers*, Oct 2006, pp. 233–237.

[6] Z. Liu, Y. Liu, and C. Li, "Distributed sparse recursive least-squares over networks," *IEEE Transactions on Signal Processing*, vol. 62, no. 6, pp. 1386–1395, March 2014.

[7] G. Mateos, I. D. Schizas, and G. B. Giannakis, "Distributed recursive least-squares for consensus-based in-network adaptive estimation," *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4583–4588, Nov 2009.

[8] G. Mateos and G. B. Giannakis, "Distributed recursive least-squares: Stability and performance analysis," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3740–3754, July 2012.

[9] R. Abdolee, B. Champagne, and A. H. Sayed, "Diffusion lms for source and process estimation in sensor networks," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*, Aug 2012, pp. 165–168.

[10] X. Zhao and A. H. Sayed, "Learning over social networks via diffusion adaptation," in *2012 Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, Nov 2012, pp. 709–713.

[11] K. Eftaxias and S. Sanei, "Diffusion adaptive filtering for modelling brain responses to motor tasks," in *2013 18th International Conference on Digital Signal Processing (DSP)*, July 2013, pp. 1–5.

[12] S. Monajemi, S. Sanei, and S. Ong, "Advances in bacteria motility modelling via diffusion adaptation," in *2014 22nd European Signal Processing Conference (EUSIPCO)*, Sep. 2014, pp. 2335–2339.

[13] P. D. Lorenzo, S. Barbarossa, P. Banelli, and S. Sardellitti, "Lms estimation of signals defined over graphs," in *2016 24th European Signal Processing Conference (EUSIPCO)*, Aug 2016, pp. 2121–2125.

[14] ——, "Adaptive least mean squares estimation of graph signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 555–568, Dec 2016.

[15] P. D. Lorenzo, P. Banelli, and S. Barbarossa, "Optimal sampling strategies for adaptive learning of graph signals," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Aug 2017, pp. 1684–1688.

[16] P. D. Lorenzo, P. Banelli, S. Barbarossa, and S. Sardellitti, "Distributed adaptive learning of graph signals," *IEEE Transactions on Signal Processing*, vol. 65, no. 16, pp. 4193–4208, Aug 2017.

[17] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "A graph diffusion LMS strategy for adaptive graph signal processing," 2017.

[18] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[19] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph Signal Processing: Overview, Challenges and Applications," *ArXiv e-prints*, Dec. 2017.

[20] S. Chen, R. Varma, A. Sandryhaila, and J. Kovacevic, "Discrete signal processing on graphs: Sampling theory," *CoRR*, vol. abs/1503.05432, 2015. [Online]. Available: http://arxiv.org/abs/1503.05432

[21] A. Gavili and X. P. Zhang, "On the shift operator, graph frequency, and optimal filtering in graph signal processing," *IEEE Transactions on Signal Processing*, vol. 65, no. 23, pp. 6303–6318, Dec 2017.

[22] I. Jaboski, "Graph signal processing in applications to sensor networks, smart grids, and smart cities," *IEEE Sensors Journal*, vol. 17, no. 23, pp. 7659–7666, Dec 2017.

[23] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 80–90, Sept 2014.

[24] W. Huang, T. A. W. Bolton, J. D. Medaglia, D. S. Bassett, A. Ribeiro, and D. V. D. Ville, "A graph signal processing perspective on functional brain imaging," *Proceedings of the IEEE*, pp. 1–18, 2018.

[25] W. Huang, L. Goldsberry, N. F. Wymbs, S. T. Grafton, D. S. Bassett, and A. Ribeiro, "Graph frequency analysis of brain signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 7, pp. 1189–1203, Oct 2016.

[26] G. Cheung, E. Magli, Y. Tanaka, and M. K. Ng, "Graph spectral image processing," *Proceedings of the IEEE*, pp. 1–24, 2018.

[27] K. Yamamoto, M. Onuki, and Y. Tanaka, "Deblurring of point cloud attributes in graph spectral domain," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 1559–1563.

[28] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovaevi, "Contour-enhanced resampling of 3d point clouds via graphs," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 2941–2945.

[29] J. Pang and G. Cheung, "Graph laplacian regularization for image denoising: Analysis in the continuous domain," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1770–1785, April 2017.

[30] A. Sandryhaila and J. M. F. Moura, "Classification via regularization on graphs," in *2013 IEEE Global Conference on Signal and Information Processing*, Dec 2013, pp. 495–498.

[31] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec 2014, pp. 872–876.

[32] ——, "Signal recovery on graphs: Variation minimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 17, pp. 4609–4624, Sept 2015.

[33] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka, "Graph signal denoising via trilateral filter on graph spectral domain," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 137–148, June 2016.

[34] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3462–3477, July 2017.

[35] X. Wang, M. Wang, and Y. Gu, "A distributed tracking algorithm for reconstruction of graph signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 728–740, June 2015.

[36] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Transactions on Signal Processing*, vol. 64, no. 14, pp. 3775–3789, July 2016.

[37] S. Chen, R. Varma, A. Singh, and J. Kovaevi, "Signal recovery on graphs: Fundamental limits of sampling strategies," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 539–554, Dec 2016.

[38] M. Tsitsvero, S. Barbarossa, and P. D. Lorenzo, "Signals on graphs: Uncertainty principle and sampling," *IEEE Transactions on Signal Processing*, vol. 64, no. 18, pp. 4845–4860, Sep. 2016.

[39] M. Wang, W. Xu, E. Mallada, and A. Tang, "Sparse recovery with graph constraints," *IEEE Transactions on Information Theory*, vol. 61, no. 2, pp. 1028–1044, Feb 2015.

[40] H. E. Egilmez and A. Ortega, "Spectral anomaly detection using graph-based filtering for wireless sensor networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 1085–1089.

[41] Y. Tanaka and A. Sakiyama, "$m$ -channel oversampled graph filter banks," *IEEE Transactions on Signal Processing*, vol. 62, no. 14, pp. 3578–3590, July 2014.

[42] S. P. Chepuri and G. Leus, "Graph sampling for covariance estimation," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 3, pp. 451–466, Sept 2017.

[43] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Aggregation sampling of graph signals in the presence of noise," in *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, Dec 2015, pp. 101–104.

[44] N. Tremblay, P. O. Amblard, and S. Barthelm, "Graph sampling with determinantal processes," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Aug 2017, pp. 1674–1678.

[45] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Transactions on Signal Processing*, vol. 65, no. 2, pp. 274–288, Jan 2017.

[46] S. Segarra, A. G. Marques, and A. Ribeiro, "Linear network operators using node-variant graph filters," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 4850–4854.

[47] X. Shi, H. Feng, M. Zhai, T. Yang, and B. Hu, "Infinite impulse response graph filters in wireless sensor networks," *IEEE Signal Processing Letters*, vol. 22, no. 8, pp. 1113–1117, 2015.

[48] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3042–3054, June 2014.

[49] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," *Applied and Computational Harmonic Analysis*, vol. 40, no. 2, pp. 260 – 291, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520315000214

[50] S. Kruzick and J. M. F. Moura, "Optimal filter design for signal processing on random graphs: Accelerated consensus," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1258–1272, March 2018.

[51] ——, "Optimal filter design for signal processing on random graphs: Accelerated consensus," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1258–1272, March 2018.

[52] S. Segarra, A. G. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 4117–4131, Aug 2017.

[53] D. B. H. Tay and Z. Lin, "Design of near orthogonal graph filter banks," *IEEE Signal Processing Letters*, vol. 22, no. 6, pp. 701–704, June 2015.

[54] G. Mateos and G. Giannakis, "Distributed recursive least-squares: Stability and performance analysis," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3740–3754, 7 2012.

[55] S. Farahmand and G. B. Giannakis, "Robust RLS in the presence of correlated noise using outlier sparsity," *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 3308–3313, June 2012.

[56] J. Chen, C. Richard, Y. Song, and D. Brie, "Transient performance analysis of zero-attracting lms," *IEEE Signal Processing Letters*, vol. 23, no. 12, pp. 1786–1790, Dec 2016.

[57] P. D. Lorenzo and A. H. Sayed, "Sparse distributed learning based on diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 61, no. 6, pp. 1419–1433, March 2013.

[58] A. S. Pranamesh Chakraborty, Chinmay Hegde, "Trend filtering in network time series with applications to traffic incident detection," *Time Series Workshop, Neural Information Processing Systems*, 2017.

[59] J. Chen and A. H. Sayed, "Distributed pareto optimization via diffusion strategies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 205–220, April 2013.

[60] Y. Chen, Y. Gu, and A. O. Hero, "Regularized Least-Mean-Square Algorithms," *ArXiv e-prints*, Dec. 2010.

[61] "National climate data center," ftp://ftp.ncdc.noaa.gov/pub/data/gsod.

[62] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, Dec 2016.

[63] H. Wang and A. Banerjee, "Online alternating direction method (longer version)," *CoRR*, vol. abs/1306.3721, 2013. [Online]. Available: http://arxiv.org/abs/1306.3721

[64] H. Ouyang, N. He, L. Tran, and A. Gray, "Stochastic alternating direction method of multipliers," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 80–88. [Online]. Available: http://proceedings.mlr.press/v28/ouyang13.html

[65] "National climate data center," ftp://ftp.ncdc.noaa.gov/pub/data/noaa/2017/.

[66] C. C. M. D. E. K. A. Lakhina, K. Papgiannaki and N. Taft, "Structural analysis of network traffic flows," 2004.

[67] R. M.-C. P. G.-T. R. T. Gabriel Maci Fernndez, Jos Camacho, "UGR'16: a new dataset for the evaluation of cyclostationarity-based network IDSs," 2017.

[68] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, Sept 2017.

[69] E. Pavez, H. E. Egilmez, and A. Ortega, "Learning graphs with monotone topology properties and multiple connected components," *IEEE Transactions on Signal Processing*, vol. 66, no. 9, pp. 2399–2413, May 2018.

[70] D. Thanou, X. Dong, D. Kressner, and P. Frossard, "Learning heat diffusion graphs," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 3, pp. 484–499, Sept 2017.

[71] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 12 2007. [Online]. Available: https://dx.doi.org/10.1093/biostatistics/kxm045

[72] S. Sardellitti, S. Barbarossa, and P. D. Lorenzo, "On the graph fourier transform for directed graphs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 796–811, Sept 2017.

[73] R. Shafipour, A. Khodabakhsh, G. Mateos, and E. Nikolova, "A Directed Graph Fourier Transform with Spread Frequency Components," *ArXiv e-prints*, Apr. 2018.

[74] G. Gloub and J. Wilkinson, "Ill-conditioned eigensystems and computation of the jordan cononical form," *SIAM Rev.*, vol. 18, no. 4, pp. 578–619, Oct 1976.

[75] V. Kalofolias, "How to learn a graph from smooth signals," *arXiv e-prints*, p. arXiv:1601.02513, Jan. 2016.

[76] S. Jung, J.-H. Seo, and P. Park, "A variable step-size diffusion normalized least-mean-square algorithm with a combination method based on mean-square deviation," *Circuits, Systems, and Signal Processing*, vol. 34, 02 2015.

[77] L. Lu and H. Zhao, "Diffusion leaky LMS algorithm: Analysis and implementation," *Signal Processing*, vol. 140, pp. 77 – 86, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168417301858

[78] J. Ni, J. Chen, and X. Chen, "Diffusion sign-error LMS algorithm: Formulation and stochastic behavior analysis," *Signal Processing*, vol. 128, pp. 142 – 149, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168416300159

[79] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, July 2017.

[80] J. Yang and S. Segarra, "Enhancing geometric deep learning via graph filter deconvolution," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Nov 2018, pp. 758–762.

[81] F. Gama, A. G. Marques, G. Leus, and A. Ribeiro, "Convolutional neural network architectures for signals supported on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 1034–1049, Feb 2019.

[82] F. P. Such, S. Sah, M. A. Dominguez, S. Pillai, C. Zhang, A. Michael, N. D. Cahill, and R. Ptucha, "Robust spatial filtering with graph convolutional neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 884–896, Sep. 2017.