Graph Normalized-LMP Algorithm for Signal Estimation Under Impulsive Noise

Yi Yan1 · Radwa Adel1 · Ercan Engin Kuruoglu1

Received: 28 February 2022 / Revised: 21 July 2022 / Accepted: 26 July 2022 / Published online: 27 August 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract

We introduce an adaptive graph normalized least mean pth power (GNLMP) algorithm that utilizes graph signal processing (GSP) techniques, including bandlimited filtering and node sampling, to estimate sampled graph signals under impulsive noise. Different from least-squares-based algorithms, such as the adaptive GSP Least Mean Squares (GLMS) algorithm and the normalized GLMS (GNLMS) algorithm, the GNLMP algorithm has the ability to reconstruct a graph signal that is corrupted by non-Gaussian noise with heavy-tailed characteristics. Compared to the recently introduced adaptive GSP least mean pth power (GLMP) algorithm, the GNLMP algorithm reduces the number of iterations to converge to a steady graph signal. The convergence condition of the GNLMP algorithm is derived, and the ability of the GNLMP algorithm to process multidimensional time-varying graph signals with multiple features is demonstrated. Simulations show that the performance of the GNLMP algorithm in estimating steady-state and time-varying graph signals is faster than GLMP and is more robust in comparison to GLMS and GNLMS.

Keywords

Graph signal processing · Impulsive noise · Alpha-stable noise · Normalized least mean pth power algorithm · Multidimensional graph signal

1 Introduction

The effectiveness of graphs in representing irregular data made graphs popular in the era of data science [1, 2]. However, with the technical advancements, we now can acquire data at a rate that is faster than ever before, faster than we can process data [1]. Research in graph signal processing (GSP) provided a solution to the problem of processing multivariate data by extending classical signal processing techniques such as Fourier transform and wavelet transform to graphs utilizing the spectral graph theory [1–6]. GSP-inspired ideas have a broad area of applications in various fields of study such as brain signals analysis [7], 5G Networks monitoring [8], temperature data modelling [9], protein-protein interaction predictions [10], and traffic events modelling [11]. By defining graph convolution in neural networks using the Graph Fourier Transform (GFT), GSP has entered the field of deep learning; architectures such as the ChebNet and the graph convolutional network (GCN) are both deep learning architectures based on GSP backbones [12, 13]. In classical signal processing, adaptive filtering algorithms are capable of performing online estimation of a time-varying signal and tracking the signal under noise. The combination of adaptive algorithms and GSP techniques first demonstrated promising performance in the adaptive Graph Least Mean Squares (GLMS) algorithm [8] at online estimation of time-varying graph signals corrupted by noise. The adaptive Graph recursive least squares (GRLS) algorithm in [14] is another GSP adaptive algorithm with faster convergence performance than the GLMS. The adaptive Graph Normalized LMS (GNLMS) algorithm [9] was proposed as an improved version with faster convergence speed than the GLMS algorithm and less computation time than the GRLS algorithm at about the same convergence performance. The introduction of adaptive algorithms to the GSP field results in parallelism between their classical signal processing counterparts.
Impulsive noise processes generally possess large or even infinite variance and the distribution generally has heavy-tailed characteristics, causing the least-squares-based algorithm to diverge and result in unstable estimation [15]. However, the least-squares-based approaches assume the data and the noise to be Gaussian, which is an oversimplification of the noise scenarios seen in real life [16, 17]. The GLMS and the GNLMS algorithms both suffer from unstable behavior under the presence of impulsive noise with diverging variance since they are least-squares-based algorithms. Existing literature have shown that ambient noise can have impulsive character in scenarios such as underwater communications [18], PLC communications [19], radar signal processing [20], and astrophysics [21]. Various distribution families were proposed in the past to model impulsive noise, including student-t, generalized Gaussian, and α-stable distributions [17]. The α-stable distribution family, which is the generalization of the Gaussian distribution, has been successfully used to model various kinds of noise and stands out due to conforming to the generalized central limit theorem [17, 22]. Applying minimum dispersion (MD) criterion by $l_p$-norm optimization instead of MMSE by $l_2$-norm optimization was suggested in the literature to avoid the problem of instability of least-squares [23–25]. This led to Graph Least Mean $p^{th}$ power (GLMP) algorithm [26], which utilizes the MD criterion and provides a robust estimation of graph signal under the presence of α-stable noise [26]. Other than the MD criterion based on $l_p$-norm optimization, in a non-graph context, there are prior works in the literature that suggest using Maximum Versoria Criterion [27] or Maximum Correntropy Criterion [28] to deal with the non-Gaussian noise. Although the GLMP algorithm is able to estimate the graph signal corrupted by α-stable noise, it suffers from a slow convergence speed similar to the GLMS algorithm.

The previously mentioned GLMS, GRLS, GNLMS, and GLMP algorithms only operate on graph signals with one single feature defined on the nodes of the graph, but data in the real world is often multidimensional and multi-featured. There are many real-life scenarios where the data is multidimensional and can potentially benefit if modeled using multidimensional graph signals. For example, in [29] multiple sensors are placed at multiple locations of the human body to monitor various vital signs and body motion, which could potentially be modeled using multi-feature GSP by treating the vital signals or the motions as the features and by constructing the graph topology using the location of the sensors. In [30], a graph-theoretic model is given for multi-channel EEG signal, but the multi-channel data is lost during the preprocessing phase by reducing the dimension to 1 using synchronization likelihood; the rest of the processing algorithm is done on the single-feature data. The GNLM algorithm in [9] is used to conduct online prediction of temperature using recordings from weather stations; it would be more beneficial if other data such as air quality, wind speed, precipitation, and humidity could be utilised simultaneously with the temperature. In [31], air pollutants recordings in weather stations, including CO, NO2, O3, PM10, PM2.5, and SO2, are monitored using a fusion of the attention mechanism and the GCN, but the inherent bulkiness of Neural Networks caused by their high complexity prohibits them to be applied to low-cost online applications. Thus, there is a need for online processing of graph signals with multiple features defined on a single graph topology at a low cost.

In this paper, we propose a novel adaptive algorithm for GSP: the graph normalized least mean $p^{th}$ power (GNLMP) algorithm. Our proposed GNLMP algorithm does not experience the instability of the GLMS or GNLMS algorithms caused by the diverging variance of impulsive noise. Instead of a fixed step-size parameter seen in the GLMP algorithm, the GNLMP algorithm uses a time-varying convergence matrix $\mathbf{M}[k]$ that significantly reduces the number of iterations to converge to a stable value. In a preliminary version of this work presented at a conference [32], a derivation of the GNLMP algorithm based on the MD criterion was introduced with the concept of spectral-domain normalization to speed up the estimation process. In this paper, we provide a more complete derivation including a steady-state convergence analysis of the GNLMP algorithm. We further extended the GNLMP algorithm to multi-feature graph signals, so that it could process multiple graph signals defined over the same graph topology simultaneously and overcome the limitation of the graph signal dimension. This simultaneous multi-feature processing of graph signals is not seen in the GLMS, GRLS, GNLMS, and GLMP algorithms. The proposed GNLMP algorithm is tested for the ability to reconstruct a sampled graph signal under various α-stable noises using synthetic and real data. The single feature steady-state experiments of [32] are redesigned to include a comparison with the theoretical performance obtained from the steady-state convergence analysis. Three experiments are provided for parameter selection of the GNLMP algorithm. In time-varying graph signal reconstruction experiments, instead of only experimenting on single-feature time-varying graph signals, additional experiments are conducted on multi-feature time-varying graph signals.

We present the notation used in this paper and some technical background information on GSP in Sect. 2. In Sect. 3, we present the GNLMP algorithm, along with the computational complexity analysis, and the steady-state convergence analysis. An extension of the GNLMP algorithm to graph signals with multiple features is also discussed in Sect. 3. Experimental studies can be found in Sect. 4. Section 5 summarizes and concludes the work.
2 Notation and Background

2.1 Notation

In this paper, the following notation will be adopted. We use \( x, \), \( \mathbf{x} \), and \( \mathbf{X} \) to denote numbers, vectors, and matrices respectively. Subscript \( x_i \) represents the \( i \)-th element of \( x \), and subscript \( \mathbf{X}_{ij} \) is the element on the \( i \)-th row and \( j \)-th column. The notation \( \mathbf{X}^T \) is the matrix transpose operation. We use the notation \( \text{support}(\cdot) \) to denote the operation of dropping the columns with all zeros. We use \( \circ \) to indicate the Hadamard product between two matrices. In this paper, single vertical bars \( |x| \) denotes the absolute value of \( x \), and \( |x|^p \) is the element-wise exponent of \( x \) where \( p \) is a real number. Double vertical bars \( ||x|| \) are for vector norms: \( ||x||_2 \) is the \( L_2 \) -norm and \( ||x||_p \) is the \( L_p \) -norm. Here we further define a notation \( \|x\|_E \) that we will use later, which indicates the square of weighted Euclidean norm \( x^T[k] \mathbf{D} x[k] \). The notation \( \mathbb{E} \) stands for the expectation. Finally, we define the \( \text{Sign}(\cdot) \) function that we are going to use extensively in later sections as

\[
\text{Sign}(x) = \begin{cases} 
0, & \text{if } x = 0, \\
\frac{x}{|x|}, & \text{otherwise}. 
\end{cases}
\tag{1}
\]

In Table 1, we provide a list of selection of symbols that will appear later in the paper along with their meanings.

2.2 Graph Signal Processing Basics

Let us first define a graph \( G = (\mathcal{V}, \mathcal{E}) \), where \( \mathcal{V} \) is the set of \( N \) nodes, and \( \mathcal{E} \) is the set of edges. For a weighted graph, the edge weight from node \( v_i \) to node \( v_j \) is the \( (i,j) \)-th entry of the graph adjacency matrix \( \mathbf{A} \), and 0 if there is no connection between two nodes. For an unweighted graph the edge weights are 1 if there is an edge between node \( v_i \) to node \( v_j \), and 0 otherwise. The degree matrix \( \mathbf{D} \) of an undirected and unweighted graph is a diagonal matrix with the \( i \)-th diagonal entry being the number of edges \( \epsilon_i \) has. The degree matrix of an undirected and weighted graph is a diagonal matrix with the \( i \)-th diagonal entry being the summation of edges weights of node \( v_i \). A graph signal \( x \) is the function values defined on the nodes of the graph \( G \); each element in \( x \) corresponds to a signal at one single node.

The graph Laplacian matrix \( \mathbf{L} \) of an undirected graph is a positive semi-definite matrix and is defined as \( \mathbf{L} = \mathbf{D} - \mathbf{A} \). The GFT is defined using the eigenvector decomposition of \( \mathbf{L} \): \( \mathbf{L} = \mathbf{U} \mathbf{D} \mathbf{U}^T \). Here the decomposition will result in \( N \) distinct eigenvalue-eigenvector pairs: the matrix \( \mathbf{A} \) is the eigenvalue matrix with \( N \) eigenvalues on the diagonals; the matrix \( \mathbf{U} \) is composed of the \( N \) orthonormal eigenvectors of \( \mathbf{L} \) [33]. The GFT of graph signal \( x \) is used to transform \( x \) from spatial-domain to spectral-domain and is defined as \( \hat{x} = \mathbf{U}^T x \), which is the projection of \( x \) onto \( \mathbf{U} \). The inverse graph Fourier transform (IGFT) \( x = \mathbf{U} \hat{x} \) transforms \( \hat{x} \) from spectral-domain to spatial domain.

GSP algorithms can benefit from sparsity defined in both the spatial domain and the spectral domain. A bandlimited graph signal is sparse in the spectral-domain [9, 33] defined by a frequency set \( \mathcal{F} \) with \( F \) spectral-domain elements. To get a bandlimited representation \( x_0 \) of a graph signal \( x \), we apply a bandlimiting filter \( \mathbf{S} \) based on \( \mathcal{F} \), where \( \mathbf{S} \in \mathbb{R}^{N \times N} \) is a diagonal matrix with idempotent and self-adjoint properties defined as \( \mathbf{S}_{ii} = 1 \) if \( i \subseteq \mathcal{F} \) and 0 otherwise. The filter \( \mathbf{S} \) is applied to a graph using graph convolution \( \mathbf{B} x \), where \( \mathbf{B} = \mathbf{U} \mathbf{S} \mathbf{U}^T \). To simplify the notation, we define \( \mathbf{U}_F^T = \text{support}(\mathbf{S}) \mathbf{U}^T \), then \( \mathbf{B} = \mathbf{U}_F^T \mathbf{U}_F \) but \( \mathbf{I} = \mathbf{U}_F^T \mathbf{I} \mathbf{U}_F \). The bandlimiteness of a graph signal \( x_0 \) with frequencies \( \mathcal{F} \) provides us the relationship \( x_0 = \mathbf{B} x_0 \) [33]. Figure 1 displays an example of a bandlimited graph signal on a random sensor graph with \( N = 50 \) nodes and \( |\mathcal{F}| = 20 \). A graph signal represented using only a few sampled nodes is sparse in the spatial domain and can be obtained based on a sampling set \( \mathcal{S} \subseteq \mathcal{V} \). The sampling operation is done by an idempotent and self-adjoint diagonal sampling matrix \( \mathbf{D}_\mathcal{S} \), with the diagonal entries equal to 1 when a node is sampled based on a sampling set \( \mathcal{S} \subseteq \mathcal{V} \) and 0 otherwise.

---

Table 1: Symbols and their meanings.

| Symbol | Meaning |
|--------|---------|
| \( \mathbf{A} \in \mathbb{R}^{N \times N} \) | Adjacency matrix of a graph |
| \( \mathbf{D} \in \mathbb{R}^{N \times N} \) | Degree matrix of a graph |
| \( \mathbf{L} \in \mathbb{R}^{N \times N} \) | Laplacian matrix of a graph |
| \( \mathbf{D}_\mathcal{S} \in \mathbb{R}^{N \times N} \) | Sampling Matrix |
| \( \mathbf{U} \in \mathbb{R}^{N \times N} \) | Eigenvector matrix of \( \mathbf{L} \) |
| \( \mathbf{U}_F \in \mathbb{R}^{N \times F} \) | Bandlimited eigenvector matrix of \( \mathbf{L} \) |
| \( \mathbf{I} \) | Identity matrix |
| \( k \) | Iteration number |
| \( \hat{x}[k] \in \mathbb{R}^N \) | Estimated graph signal at step \( k \) |
| \( \hat{x}[k] \in \mathbb{R}^N \) | Estimation error at step \( k \) |
| \( \hat{x}[k] \in \mathbb{R}^N \) | GFT of \( x[k] \) |
| \( \omega[k] \in \mathbb{R}^N \) | Noise of the graph signal |
| \( \hat{x}[k] \in \mathbb{R}^N \) | Bandlimited graph signal |
| \( \hat{x}[k] \in \mathbb{R}^N \) | Noisy observation of \( x_0 \) |
| \( \hat{x}[k] \in \mathbb{R}^N \) | GFT of \( x[k] \) |
| \( \mu \) | GLMS step size |
| \( \mu \) | GLMP step size |
| \( \mu \) | GNLM step size |
| \( \mu \) | GNLM step size |
| \( \mathbf{M}_0 \in \mathbb{R}^{F \times F} \) | GNLMs convergence matrix |
| \( \mathbf{M}_0 \in \mathbb{R}^{F \times F} \) | GNLM convergence matrix |
| \( \mathbf{X} \in \mathbb{R}^{N \times d} \) | Multi-feature estimated graph signal |
| \( \mathbf{W} \in \mathbb{R}^{N \times d} \) | Noise of multi-feature graph signal |
| \( \mathbf{M}_\mu \in \mathbb{R}^{N \times N} \) | Multi-feature GNLM step sizes |
2.3 The $\alpha$-stable Distribution

To model impulsive noise in accordance with previous literature [20], we use the symmetric $\alpha$-stable distribution (SaS), which is a generalization of the Gaussian distribution. The SaS distribution is governed by three parameters: the characteristic exponent $\alpha$ that acts as tail-shape parameter, the dispersion $\gamma$ that acts as scale factor, and the location parameter $\delta$. The SaS distribution obeys the generalised central limit theorem where linear combinations of independent SaS random variables belong still to SaS. The parameter $\alpha$ controls the impulsiveness of SaS; $\delta$ is the mean when $1 < \alpha \leq 2$ and the median when $1 < \alpha < 2$. The parameter $\gamma$ controls the deviation around the mean or median. Unless when $\alpha = 2$, the variance of SaS diverges. The SaS distributions have no analytic PDF except when $\alpha = 1$ and for $\alpha = 2$, which are the Cauchy distribution and the Gaussian distribution respectively. However, the characteristic function of SaS could be expressed analytically as

$$
\phi(t) = \exp \left[ j\delta t - \gamma |t|^{\alpha} \right].
$$

The $l_p$-norm optimization used in our GNLMP algorithm utilizes the minimization of the dispersion, which is equivalent to minimizing the $p^{th}$-order moment when $1 < \alpha < 2$, or the fractional lower order moment (FLOM) [17]

$$
\text{FLOM}(p, \alpha, \gamma) = \mathbb{E}[|X|^p] = C(p, \alpha)\gamma^{p/\alpha},
$$

with $C(p, \alpha) = \frac{2^{p+1} \Gamma \left( \frac{p+1}{2} \right) \Gamma \left( -\frac{p}{\alpha} \right)}{\alpha \sqrt{\pi} \Gamma \left( -\frac{p}{2} \right)}$

where $\mathbb{E}$ is the expectation operation.

3 Adaptive GNLMP Algorithm for GSP

3.1 GLMS, GNLMS, and GLMP Analyses

Following the convention of the previous adaptive GSP algorithms, we consider a bandlimited graph signal $x_0$, and its noisy observation with missing node values at iteration $k$ to be expressed as a sampled noisy graph signal $y[k] = D_S(x_0 + w[k])$. In this paper, the noise $w[k] \in \mathbb{R}^N$ is modeled using $\alpha$S with $\alpha \in (1, 2)$ on each node, and the noises among individual nodes are i.i.d. The case where $\alpha \leq 1$ is not considered because $\alpha \leq 1$ indicates highly impulsive behavior and is rarely observed [26].

Least-squares-based algorithms are used extensively due to their simplicity of implementation. At iteration $k$, using the current step estimate $\hat{x}[k]$, the cost function $J(\hat{x}[k])$ for GLMS minimizes the mean-squared error of the estimation [8]:

$$
J(\hat{x}[k]) = \mathbb{E}[f(\hat{x}[k])],
$$

where $f(\hat{x}[k]) = \|y[k] - D_S\hat{B}[k]\|^2_2$.

Using the cost function in (4) and the bandlimitedness property of $x_0$ and $\hat{x}[k]$, a convex optimization problem can be formulated as below:

$$
\min_{\hat{x}[k]} J(\hat{x}[k])
$$

s.t. $B\hat{x}[k] = \hat{x}[k]$.

The optimized solution of (5) could be obtained using a stochastic gradient based method. Knowing that $B\hat{x}[k] = \hat{x}[k]$ for bandlimited graph signal, the spatial-domain update function of the GLMS algorithm is

$$
\hat{x}[k+1] = \hat{x}[k] - \frac{\mu_{\text{lms}}}{2} \frac{\partial f(\hat{x}[k])}{\partial \hat{x}[k]}
$$

$$
= \hat{x}[k] + \mu_{\text{lms}} B_D S(y[k] - \hat{x}[k]).
$$

A step-size variable $\mu_{\text{lms}}$ is added to the GLMS algorithm to control the amount of the update of each iteration. Although simple to implement, the GLMS algorithm has two major drawbacks. First, the GLMS algorithm takes many iterations to convergence to a steady value [9]. Second, the GLMS algorithm is derived with the assumption that noise follows the Gaussian distribution, but in reality, there are various non-Gaussian noise scenarios [17–20] that GLMS cannot handle.

In classical adaptive filtering, one possible solution to increase the convergence speed of the LMS algorithm is the normalization operation [34]. Although in GSP the analogue of the classical NLMS is the GNLMS algorithm, instead of just having a fixed step size, the GNLMS algorithm includes a symmetric convergence matrix as normalization [9]. The update function of the GNLMS algorithm could be expressed as

$$
\hat{x}[k+1] = \hat{x}[k] + \mu_{\text{lms}} U_{f} D_S U_{f}^T(y[k] - \hat{x}[k]),
$$

where $M_n = (U_{f} D_S U_{f})^{-1}$.
3.2 GLNMP Algorithm Derivation

Even though the GLMP algorithm is able to estimate the graph signal corrupted by SαS, the GLMP algorithm still does not solve the problem of slow convergence speed compared to the GLMS algorithm. We propose the GLNLP algorithm combining also the advantage of normalisation which was showed for classical adaptive algorithms to speed up convergence and to make the estimates more stable.

We first transform (8) into the spectral-domain using GFT and replace $\mu_{ln}$ with a time-varying convergence matrix $M[k]$, leading to the spectral-domain update of the GLNLP algorithm:

$$\hat{\mathcal{S}}[k + 1] = \hat{\mathcal{S}}[k] + M[k]U_F[e[k]]\varrho^{-1} \text{Sign}(e[k]),$$

where $e[k] = D_S(y[k] - \hat{X}[k])$ is the current step spatial-domain estimation error.

In order to find $M[k]$, following the derivations in [9], we define the a posteriori error $e[k] = D_S(\hat{X}[k] - U_x \hat{X}[k + 1])$ to measure the error between current step prediction and the next step prediction. We also define $\Delta \hat{\mathcal{S}}^n = ||e[k]||_\varrho^n - ||e[k]||_\varrho$ to be a measure of the closeness of $e[k]$ to $e[k]$. In GLNLP, in addition to the optimization problem in (5), which can be seen as minimizing $e[k]$, we want $\Delta \hat{\mathcal{S}}^n$ to be minimized as well. This minimization of $\Delta \hat{\mathcal{S}}^n$ with respect to $M[k]$ could be interpreted as using the spectral domain difference between $\hat{\mathcal{S}}[k + 1]$ and $\hat{\mathcal{S}}[k]$ as spectral domain normalization of the update term [9], while keeping $e[k]$ small based on (4). Taking the derivative of $\Delta \hat{\mathcal{S}}^n$ with respect to $M[k]$, we have:

$$0 = \Omega^T(B[e[k]]\varrho^{-1} \text{Sign}(e[k])) + (e[k])\varrho^{-1} \text{Sign}(e[k])B)\Omega,$$

where $\Omega = D_S U_F M[k] U_F^T(e[k]||\varrho^{-1} \text{Sign}(e[k])) - e[k].$

Notice in (10) that the two sides of the addition are transposes of each other. Therefore, we can solve (10) by setting $\Omega$ to zero. Utilizing the property $I = U_F^T U_F$ and noticing that $D_S$ is idempotent and self-adjoint, the expression for $M[k]$ is

$$M[k] = (U_F^T D_S \text{diag}(e[k] - \hat{X}[k])^p U_F)^{-1}. (11)$$

Following the convention of classical adaptive filtering, we add a step size parameter $\mu$ to balance the convergence speed of the algorithm and the effectiveness of the update at each step. The spectral-domain update step of the adaptive GLNLP algorithm can be formalized as

$$\hat{\mathcal{S}}[k + 1] = \hat{\mathcal{S}}[k] + \mu M[k][U_F^T(e[k])\varrho^{-1} \text{Sign}(e[k])),$$

where $M[k]$ is shown in (11). Notice that this update function not only adaptively updates based on the error, but also has a time-varying parameter $M[k]$, which is different from the GLMS, the GNLMS, and the GLMP algorithms as they only adaptively update based on the error. It is worth mentioning that when we set $\mu = 2$ in (12), $M[k]$ will reduce to $M_n$, and $|e[k]||\varrho-1 \text{Sign}(e[k])$ reduces to just $e[k]$, which is exactly the update function of GNLMS in (7).

3.3 Approximation of GLNLP

Compared to the GLMP algorithm, the extra computations for calculating $M[k]$ increases the run time of the GLNLP algorithm. We will discuss the computational complexity of our GLNLP algorithm later in Sect. 3.4. Before that, we would like to simplify our algorithm without sacrificing much of the performance.

Since the problem is set up as convex optimization, it is safe to assume that after a few iterations the estimation error is mainly caused by the noise. Even though we do not have an analytical PDF for the SαS, we can use the FLOM from (3) to obtain noise statistics. Also, the sampling strategy we adopted from [9] does not change as the algorithm progresses. Using these facts, we can approximate $M[k]$ to be $(U_F^T D_S R U_F)^{-1}$, with $R = (E[w[k]]\varrho)^p I$ from (3). Using this approximation, we can combine all the time-independent terms of the GNLP algorithm to form a matrix to perform the spectral domain filtering and normalization.

$$B_n = U_F(U_F^T D_S R U_F)^{-1} U_F^T.$$ Now the update function of GNLP in (12) is approximated to be
\[ \dot{x}[k + 1] = \dot{x}[k] + \mu B_s( |e[k]|^{p-1} \circ \text{Sign}(e[k]) ). \] (13)

This formulation significantly reduces the number of operations done by our proposed GNLMP algorithm because by predefining \( B_n \) because \( B_n \) is calculated only once. However, (13) lacks the time-variability of \( M[k] \) that can be seen in (12). To maintain the adaptiveness gained from the time-variant \( M[k] \) in (12) as well as maintaining the efficiency of (13), we use a threshold-based switching between (12) and (13). At step \( k \), if the total amount of update magnitude at all sampled nodes is smaller than a certain threshold, we switch from (12) to (13), otherwise the update is (12). Earlier in this section, we assumed that after a few iterations, \( e[k] \) is dominated by \( w[k] \), we set the threshold to be \( \text{threshold} = |S| \text{FLOM}(p - 1, \alpha, \gamma) \), where \(|S|\) is the cardinality of \( S \). The choice of threshold is not a strict requirement; it can be changed to other values to suit the need of the target application. The resulting GNLMP algorithm is shown in Algorithm 1.

**Algorithm 1 Threshold based GNLMP**

1. \( B_n = U_F (U_F^T D_S R U_F)^{-1} U_F^T \)
2. \( \text{threshold} = |S| \text{FLOM}(p - 1, \alpha, \gamma) \)
3. \( \text{while} \) within the iteration limit do
4. \( \text{if} \ \text{sum}(|e[k]|^{p-1}) < \text{threshold} \) then
5. \( \text{update} \ \hat{x}[k+1] \) based on (13)
6. \( \text{else} \)
7. \( \text{update} \ \hat{x}[k+1] \) based on (12)
8. \( \text{end if} \)
9. \( \text{end while} \)

### 3.4 Computational Complexity Analysis

In this section, we will analyze the computational complexity of the proposed GNLMP algorithm. Comparing the approximated GNLMP update (13) with (8), notice that both \( B_n \) and \( B \) have the same dimension \( \mathbb{R}^{N \times N} \). Also, \( D_S( |y[k] - \hat{x}[k]|^{p-1} \circ \text{Sign}(y[k] - \hat{x}[k]) ) = |e[k]|^{p-1} \circ \text{Sign}(e[k]|^{p-1}) \), the difference of two expressions is only in the notation. As a result, the GNLMP approximation in (13) has the same computational complexity as the GLMP update in (8). As for the update function in (12), the extra computations compared to (8) are the following: one diagonal matrix multiplication, one element-wise exponent, one matrix inverse, and three matrix multiplications. Matrix multiplication and matrix inverse are both \( O(N^3) \) operations, and the element-wise exponent takes only \( O(N \log(N)) \). Due to the extra computations, the actual run time of (12) might be longer than (8), but both the GLMP and the GNLMP are in fact dominated by the \( O(N^3) \) operations.

### 3.5 Steady-state Convergence Behavior of GNLMP

We would like to investigate the steady-state behavior of the GNLMP algorithm. For simplicity, we only analyze the approximated GNLMP update function in (13). In reality, \( y[k] - \hat{x}[k] \) is the error of the estimation, which is rarely zero. Then, for \( \text{Sign}(y[k] - \hat{x}[k]) \neq 0 \), (13) can be written as

\[ \dot{x}[k + 1] = \dot{x}[k] + \mu B_s D_S R_p (y[k] - \hat{x}[k]), \] (14)

where \( R_p = |\text{diag}(y[k] - \hat{x}[k])|^{p-2} \). Let the error update between \( x_0 \) and \( \hat{x}[k] \) be \( x[k] = \hat{x}[k] - x_0 \), then the error update function can be expressed as

\[ \dot{x}[k + 1] = \dot{x}[k] + \mu B_s D_S R_p (w[k] - \hat{x}[k]) \]

\[ = (I - \mu B_s D_S R_p) \dot{x}[k] + \mu B_s D_S R_p w[k]. \] (15)

We can obtain the squared error of each update step based on (15), which leads to

\[ E[\|\dot{x}[k+1]\|^2_2] \]

\[ = E[\|\dot{x}[k]\|^2_2] + \mu^2 E[\|B_s D_S R_p\|^2_2]. \] (16)

In (16), \( \Phi = (I - \mu B_s D_S R_p)^T (I - \mu B_s D_S R_p) \). Looking back into (15), we could rewrite it into the following recursive relationship:

\[ \dot{x}[k + 1] = (I - \mu B_s D_S R_p)^k \dot{x}[0] + \sum_{i=0}^{k} (I - \mu B_s D_S R_p)^k w[i]. \] (17)

As previously stated, we assume that the noise effect will dominate the error behavior and the noise is i.i.d. among the nodes, so \( R_p \) can be approximated to be \( |w[k]|^{p-2} \). We can now rewrite (16) in a recursive form:

\[ E[\|\dot{x}[k+1]\|^2_2] \]

\[ = E[\|\dot{x}[0]\|^2_2] + \mu^2 \sum_{i=0}^{k} E[\|I - \mu B_s D_S R_p\|^2_2] \]

\[ = E[\|\dot{x}[0]\|^2_2] + \mu^2 \sum_{i=0}^{k} \text{Tr} (\Phi^2 G) \] (18)

\[ = E[\|\dot{x}[0]\|^2_2] + \mu^2 \sum_{i=0}^{k} \text{vec}(G) \text{vec}(\Phi^2), \] (19)

where \( G = B_n D_S E[|w[k]|^{2p-2} D_S^2 R_s] \) and \( G' = B_n D_S E[|w[k]|^{2p-2} D_S^2 B_n] \). Equation (19) is obtained from (18) by using the Trace property \( E[\{P^T Q\}^T] = \text{Tr}(E[\{P^T Q\}]) \). Equation (20) is obtained by using the property \( \text{Tr}(QP) = \text{vec}(P^T) \text{vec}(Q) \).
where \( \text{vec}(\cdot) \) is the operation of stacking each column of a matrix into a single column vector.

In order for \( \mathbb{E}[\|\mathbf{x}[k+1]\|^2] \) to converge in steady-state estimation of a graph signal, the condition \( \|\{(I - \mu \mathbf{B}_n \mathbf{D}_S \mathbf{E}[\mathbf{R}_p]\}\} < 1 \) should be satisfied. Then in (20) the term \( \mathbb{E}[\|\mathbf{x}[0]\|_{\mathbf{P}_0}^2] \) will be 0 if \( \mathbf{x}[0] \) is bounded, and the summation term \( \mu^2 \sum_{k=0}^{\infty} \text{vec}(\mathbf{G}^T)\text{vec}(\Phi^k) \) will be a converging geometric series. Given a symmetric matrix \( \mathbf{Z} \), we have \( \|\mathbf{Z}\| = |\lambda_{\text{max}}| \), where \( \lambda_{\text{max}} \) is the largest eigenvalue of \( \mathbf{Z} \). In the case of \( \mathbf{Z} = \{(I - \mu \mathbf{B}_n \mathbf{D}_S \mathbf{E}[\mathbf{R}_p]\}\) we want \( \|\mathbf{Z}\| < 1 \) for the GNLMP algorithm to converge. Since the only user defined variable is \( \mu \), the following condition should be satisfied:

\[
0 < \mu < \frac{2}{\lambda_{\text{max}}}, \tag{21}
\]

where \( \lambda_{\text{max}} \) is the maximum eigenvalue of \( \mathbf{Z} \).

When condition (21) is satisfied, the error or the deviation in mean-squared sense of a steady-state estimation converges and can be obtained based on (20) using the property \( \text{vec}((\mathbf{Q}^T \otimes \mathbf{O})\text{vec}(\mathbf{P})) = \text{vec}(\Phi)\text{vec}(\mathbf{Q})\text{vec}(\mathbf{O}) = \text{vec}((\mathbf{Q}^T \otimes \mathbf{O})\text{vec}(\mathbf{P})) \):

\[
\lim_{k \to \infty} \mathbb{E}[\|\mathbf{x}[k]\|^2] = \mu^2\text{vec}(\mathbf{G}^T)(I - \mathbf{Z}^T \otimes \mathbf{Z})^{-1}\text{vec}(I). \tag{22}
\]

The terms \( \mathbb{E}[\|\mathbf{w}[k]\|^{2p-2}] \) in \( \mathbf{G} \) and \( \mathbb{E}[\|\mathbf{R}_p\|] \) in \( \mathbf{Z} \) can be calculated using (3).

### 3.6 Processing Multi-feature Graph Signal Using GNLMP

Let us consider a bandlimited graph signal with \( d \) features, where each feature is represented by a \( N \) by 1 vector. The graph signal of interest \( \mathbf{X}_0 \) is a matrix of size \( N \) by \( d \), with each column being one feature. An illustration of such graph signal with \( N = 197 \) nodes and \( d = 2 \) features is shown in Fig. 2. The underlying graph topology is constructed using 7-nearest neighbors approach of the geographic locations of 197 weather stations across the U.S. [35]. The multi-feature time-varying graph signal consists of two features, each representing the temperature in Fig. 2a and the wind speed Fig. 2b recorded for \( t = 95 \) hours by the weather stations.

Even though such a multi-feature graph signal can be viewed as having two graph signals defined over the same graph topology, estimation of a multi-feature graph signal using the adaptive GSP algorithms was not discussed in the literature of GLMS, GRLS, GNLS, or GLMP. One can try to treat different features as separate graph signals and process them independently. For example, the GLMS algorithm will require each feature to be treated as a single one-dimensional graph signal, thus the estimation of \( d \) features will be done using \( d \) separate runs. We would like to expand the GNLMP algorithm to process such multidimensional graph signal \( \mathbf{X}_0 \) so all features are processed online simultaneously, instead of processing \( d \) one dimensional \( \mathbf{x}_n \)'s separately. For simplicity, in this section, we consider only the approximated GNLMP shown in (13).

Let the noisy observation as iteration \( k \) of \( \mathbf{X}_0 \) be \( \mathbf{Y}[k] \), with the missing nodes modeled by sampling \( \mathbf{D}_S \). As a result, we have \( \mathbf{Y}[k] = \mathbf{D}_S(\mathbf{X}_0 + \mathbf{W}[k]) \), where \( \mathbf{W}[k] \) is the noise i.i.d. among the nodes and features. We want to minimize the error of the estimation \( \mathbf{X}[k] \) using the MD criterion. The multi-feature GNLMP update function could be obtained by solving the following optimization problem:

\[
\min_{\mathbf{X}[k]} J(\mathbf{X}[k]) \tag{23}
\]

s.t. \( \mathbf{B}_n \mathbf{X}[k] = \mathbf{X}[k] \).

where \( J(\mathbf{X}[k]) \) is the cost function

\[
J(\mathbf{X}[k]) = \mathbb{E}[\|\mathbf{Y}[k] - \mathbf{D}_S \mathbf{B}_n \mathbf{X}[k]\|_{\mathbf{P}}^p]. \tag{24}
\]

Using the stochastic gradient approach similar to the single feature case, the expression for the multi-feature GLMP update function is
\[ \hat{X}[k+1] = \hat{X}[k] + B_{\mu}(E[k])^p \text{Sign}(E[k])M_{\mu}, \]  

(25)

where \( E[k] = D_S(Y[k] - \hat{X}[k]) \) and the \( p - 1 \)th power is done element-wise. The matrix \( M_{\mu} = \text{diag}(\mu_1, \ldots, \mu_d) \) is a step size matrix so each feature has a step size that can be tuned based on user preferences. Notice that when we process a multi-feature graph signal using the multi-feature GNLMP algorithm in (25), the estimation of \( d \) features is done simultaneously in an online fashion.

When comparing (25) to (13), we can see that only \( B_{\mu} \) and \( D_S \) are unchanged. \( D_S \) depends on the sampling strategy, which is only related to the graph topology. As for \( B_{\mu} \), it depends on the eigenvector and eigenvalues of the Laplacian matrix, which ultimately is related only to the graph topology. The univariate step size \( \mu \) in (13) becomes the step size matrix \( M_{\mu} \) in (25) to facilitate multi-dimensional update behavior tuning. The vectors \( \hat{X}, Y, \) and \( e \) are changed to rectangular matrices \( \hat{X}, Y, \) and \( E \) to reflect \( d \) feature dimensions.

4 Experimental Results and Discussion

4.1 Experiment Setup

The GNLMP algorithm shown in Algorithm 1 is evaluated in Mean-squared deviation (MSD) performance under different experimental settings. For each experiment, we repeat 100 independent runs and present the results here. The instantaneous MSD at step \( k \) is

\[ \text{MSD}[k] = \mathbb{E}||\hat{x}[k] - x_0||_2^2, \]  

(26)

where the mean is taken for the 100 runs. In Sects. 4.2, 4.3, and 4.4, the experiments are conducted using a random sensor graph generated by Python PyGSP with \( N = 50 \) nodes. The generated instance used in this paper is shown in Fig. 1. The frequency bands are \( |F| = 20 \), selected using the techniques seen in [9] to maximize spectral information, and the sampling strategy is the greedy strategy used in [9] with \( |S| = 30 \).

In Sect. 4.5, a more realistic experiment is conducted by estimating a real time-varying graph signal of hourly weather collected from weather stations across the U.S. [35]. The graph signal is based on the graph topology in Fig. 2a with \( N = 197 \) and \( t = 95 \) time steps.

In Sect. 4.6, a more challenging graph signal estimation is attempted. We would like to simultaneously estimate the two features of a real multidimensional time-varying graph signal; the \( d = 2 \) features are hourly temperature and average hourly wind speed collected from weather stations across the U.S. [35]. In Fig. 2, an illustration of one time step and the \( d = 2 \) features of the graph signal are shown. For both experiments on real data, we use the greedy sampling technique in [9] with \( |S| = 130 \) to simulate missing node values and spatial domain sparsity. The frequency bands are selected using \( |F| = 125 \) bands that have the maximum spectral information to provide spectral-domain sparsity [9].

The following baseline algorithms will be selected to compare with the GNLMP algorithm based on the nature of the experiment: the GLMS algorithm, the GNLMS algorithm, and the GLMP algorithm. In all experiments, \( p = a - 0.05 \) in the GNLMP algorithm and the GLMP algorithm as suggested in [15]. The experiments were conducted on a computer with AMD Ryzen 5 3600 as CPU and RAM size of 32.0 GB using MATLAB version 2020b.

4.2 GNLMP Under Different Parameter Settings

In this section, we want to verify that the GNLMP algorithm shown in Algorithm 1 works as intended. We will conduct three experiments with the graph signal shown in Fig. 1, where the graph signal is sampled as discussed in Sect. 4.1 and is corrupted by SarS noise with \( a = 1.5 \) and \( \gamma = 0.1 \). A comparison will be conducted between the theoretical MSD and the experimental MSD under different sizes of \( \mu \) of the GNLMP algorithm. Then, we would like to investigate what Algorithm 1 gained and lost from the approximation in (13). Finally, we will observe the effect of different choices of the parameter \( \mu \) on the GNLMP algorithm compared to the GLMP algorithm under SarS noise; the observed behavior will provide us an idea of parameter selection for later experiments.

We first run the GNLMP algorithm that is based on Algorithm 1 on a graph signal reconstruction task using step sizes \( \mu = 0.05, 0.1, \) and \( 0.005 \). The MSD of the reconstruction is shown in Fig. 3 along with the theoretical MSD calculated using (22). From Fig. 3, we can conclude that as \( \mu \) decreases, the algorithm will result in lower MSD, but needs more iterations to converge to a steady value. In other words, the convergence speed of GNLMP and the effectiveness of the update at each step can be tuned by tuning the step size \( \mu \).

In Fig. 3 the theoretical MSD matches the experiment MSD when the algorithm converges to a steady estimation.

To confirm that the threshold based switching in Algorithm 1 performs as expected and to investigate how Algorithm 1 is different from purely updating using (12) or (13), we run the GNLMP algorithm using three different thresholds: \( 0, |S|F\text{LOM}(p - 1, a, \gamma), \) and \( \infty \). By setting the threshold to 0, we are essentially letting the GNLMP algorithm update only using (12), and a threshold of \( \infty \) means the GNLMP algorithm will update only using (13). The MSD results of this experiment are shown in Fig. 4. Comparing the three MSD results, (12) has the lowest MSD but takes most of the iterations to converge. This meets our expectation where (12) provides a more accurate estimation than

[Page 32]
Looking at Algorithm 1, we can see the change of the slope in the MSD, which indicates that the algorithm switched from (12) to (13) based on the threshold. It is worth mentioning that Algorithm 1 takes the least amount of iterations among the three update strategies, and has the same accuracy as (13). This is because Algorithm 1 benefits from the accurate estimation of (12) before switching to (13), and then takes advantage of the fast convergence speed of (13). By inspecting Fig. 4, the choice of setting threshold $S = \text{FLOM}(p - 1, \alpha, \gamma)$ in Algorithm 1 seems to be reasonable because this choice gives the optimal run speed and does not sacrifice the accuracy.

Knowing how the choice of $\mu$ controls the behavior of our GNLMP algorithm, we further conduct an experiment where we set $\mu_{\text{imp}} = 0.05, 0.01, 0.005$ for the GLMP, and $\mu = 0.05, 0.01, 0.005$ for the GNLMP algorithm. In this experiment, the goal is to observe whether there is a behavior difference between the GNLMP and GLMP algorithms when setting the step sizes to the same magnitude. The MSD results are shown in Fig. 5. It can be clearly observed from Fig. 5 that for the same step size, the GLMP and GNLMP algorithm behaves differently. Under the same step size, the GLMP algorithm may be slightly more accurate in terms of the MSD but takes significantly more iterations. Due to this behavior difference, in later experiments when we will compare these two algorithms under steady-state estimation, the step sizes will be chosen such that the graph signal estimation results from GLMP algorithm and the GNLMP algorithm have approximately the same MSD. This strategy will allow us to compare the run time and the number of iterations to converge the GLMP and the GNLMP algorithms under a fair condition.

### 4.3 Graph Signal Estimation Under Various $\text{S\alpha S}$ Noises

The GNLMP algorithm is compared with the GLMP algorithm in order to find out the effectiveness of the GNLMP algorithm at estimating a graph signal corrupted by various types of $\text{S\alpha S}$. We applied four different $\text{S\alpha S}$ noises for $w[k]$, with the parameters $\alpha = 2, 1.8, 1.5, 1.2$, and $\gamma = 0.1$. The step size of the GLMP algorithm is set to $\mu_{\text{imp}} = 0.005$ for all 4 noises, and for GNLMP we used grid search to find the $\mu$ values so that under each noise setting, both GNLMP and GLMP will have approximately the same MSD values. This choice of parameter selection eliminates the behavior differences between the step sizes and fairly compares the number of iterations for the algorithms to converge to the same MSD. The number of iterations is set to 6000. The MSD of the estimation of the graph signal is shown in Fig. 6 and a table of run time for different settings is summarized in Table 2.

The steady convergence behavior of the MSD in Fig. 6 indicates the GNLMP algorithm is able to stably estimate the graph signal corrupted by various types of $\text{S\alpha S}$. Looking at the run time in Table 2, we see that the GNLMP algorithm takes approximately the same amount of time as the GLMP algorithm to complete 6000 iterations. But to achieve the same MSD value the GLMP algorithm converges slower than the GNLMP algorithm, with 2 times the iterations GNLMP algorithm needed to reach the same MSD as shown in Fig. 6.
4.4 Steady-state Performance of GNLMP Algorithm

To measure steady-state graph signal reconstruction quality of the GNLMP algorithm, the graph signal in Fig. 1 is corrupted by SauS noise with parameters \( \alpha = 1.5 \) and \( \gamma = 0.1 \). The GNLMP algorithm is compared with the GLMS algorithm, the GLMP algorithm, and the GNLMS algorithm. The step size parameters are set to be \( \mu_{\text{lm}} = \mu_{\text{mp}} = 0.05 \), and \( \mu = 0.028 \). The reason behind this choice is we found using grid search that both the GLMP and GNLMP have the same MSD at \( \mu_{\text{mp}} = 0.05 \) and \( \mu = 0.028 \), which enable us to conveniently compare the number of iterations and the run time for both algorithms to converge. As for the GLMS and GNLMS, some preliminary experiments have shown that these two algorithms do not converge under SauS noise, so we set the step sizes to be the same as GLMP and GNLMP respectively. Figure 7 displays the MSD performance of estimating the graph signal along with the theoretical MSD of the GNLMP algorithm.

It can be observed from Fig. 7 that both the GLMS and the GNLMS algorithms experience rapid MSD change and are unable to converge. This unstable behavior of the GLMS algorithm and of the GNLMS is caused by the instability of using least-squares minimization under SauS noise [17]. The MD criterion used in the GNLMP algorithm does not suffer from the poor performance caused by the impulsiveness of SauS noise. The GLMP algorithm and the GNLMP algorithm finish the 400 iterations in 0.0086 seconds and 0.0092 seconds respectively. Notice in Fig. 7, the GNLMP algorithm reaches a steady MSD at iteration 150, whereas the GLMP algorithm reaches the same steady MSD at iteration 300, which confirmed that the GNLMP algorithm uses a fewer number of iterations to converge to a stable estimation.

4.5 Time-varying Graph Signal Estimation using GNLMP

Extending the idea of steady-state graph signal reconstruction, we apply the GNLMP algorithm to a time-varying graph signal of hourly temperature gathered from the real world. There are two challenges in this experiment compared to the previous experiments. First, this experiment is now time-varying, which tests the ability of the GNLMP algorithm to conduct online estimation. Second, the data used in this experiment is gathered from the real world, which makes it not a perfectly bandlimited signal. In Fig. 2a, an illustration of a single time steps of the graph signal of hourly temperature is shown. The graph topology is generated using the approach as in [9], which could be summarized as treating each station as a node and connecting each station with its 7-nearest-neighbors. The edge connections are calculated using the geographical locations of the stations. Each weather station is represented as one node on the graph and the temperature recordings \( x_{i}[k] \in \mathbb{R}^{N \times 1} \) is the graph signal of interest. In this experiment, we only consider one single feature, the hourly temperature, out of the two features available so that we are able to compare our algorithm with the baseline algorithms.

To make the comparison between GLMP and GNLMP fair, both algorithms are tuned using grid search. As for the GLMS and the GNLMS, even though both algorithms have instabilities under SauS, they could still track the time-varying signal with limited accuracy. The resulting step sizes are \( \mu_{\text{lm}} = 0.8 \), \( \mu_{\text{mp}} = 0.8 \), \( \mu_{\text{mp}} = 1.6 \), and \( \mu = 0.55 \). In this experiment, the GNLMS algorithm is being compared to the GLMS algorithm, the GLMP algorithm, and the GNLMS algorithm. The SauS noise is with parameters \( \alpha = 1.5 \) and \( \gamma = 0.1 \). Figure 8 shows a reconstruction result of one selected node of the time-varying graph signal.

| Table 2 Run Time Comparison Between GLMP and GNLMP. |
|-----------------|-----------------|-----------------|-----------------|-----------------|
|                 | \( \alpha = 2 \) | \( \alpha = 1.8 \) | \( \alpha = 1.5 \) | \( \alpha = 1.2 \) |
| GLMP            | 0.1339(s)       | 0.1330(s)       | 0.1346(s)       | 0.1372(s)       |
| GNLMP           | 0.1330(s)       | 0.1363(s)       | 0.1397(s)       | 0.1477(s)       |
Similar to the steady-state experiments, the reconstructions using the GLMS algorithm and the GNLMS algorithm are also unstable under time-varying settings due to diverging second-order moments under $S_t$ noise. This instability can be observed both in the signal estimation in Fig. 8.

4.6 Multi-feature Time-varying Graph Signal Estimation Using GNLMP

In this section, we further extend the time-varying graph signal estimation to multi-feature time-varying graph signal estimation. Figure 2 shows the two features, the hourly temperature and hourly wind speed, of the multi-feature graph signal at the time point $k = 1$. Each weather station is represented as one node on the graph, resulting in a graph signal in the form $X_0[k] \in \mathbb{R}^{d \times N \times t}$. The graph signal is estimated using the GNLMP algorithm where the estimation results for both features are outputted simultaneously. The experiment setup is essentially the same as Sect. 4.5, with $|S| = 130$ and $|F| = 125$. To take into account the fact that the two features are in different scales, two step sizes in $\mu$ of (25) are set to $\mu_1 = 0.55$ and $\mu_2 = 0.475$ respectively so that the algorithm has the best reconstruction quality. It is worth mentioning that the choice of $\mu_1 = 0.55$ is the same value as the $\mu$ parameter in Sect. 4.5 because both experiments are processing the same temperature feature on the same node. The noise parameters of the $S_t$ noise is $\alpha = 1.5$ and $\gamma = 0.1$. Figure 9 shows a reconstruction of both features at one selected node.

By inspecting Fig. 9, we see that even though each one of the two features experience different magnitudes of change, the estimation for both features are accurate under $S_t$ noise. This indicates that the step size matrix $M_\mu$ gives the GNLMP algorithm the freedom to update each feature with a different magnitude. From Fig. 9 we can see that the GNLMP algorithm is able to track the changes in both features in the multi-feature time-varying graph signal; the additional dimension of feature does not hinder the ability of the algorithm in making online predictions.

5 Conclusions

In this paper, we introduced and analyzed the adaptive GNLMP algorithm. To cope with the presence of $S_t$ noise, the GNLMP algorithm was derived based on the MD criterion. Compared to the traditional least-squares approaches, the proposed GNLMP algorithm does not suffer from the unstable estimation of least-squares caused by the heavy-tailed behavior of $S_t$ noise. The usage of a time-varying convergence matrix $M[k]$ instead of a fixed step-size parameter makes it possible for the GNLMP algorithm to use fewer iterations to converge to a stable value than the recent GLMP algorithm. For steady-state estimations, the convergence condition for the GNLMP algorithm is provided. The GNLMP algorithm was also expanded to handle multi-feature graph signals instead of graph signals with only one feature. Experimenting with sampled and bandlimited graph signals corrupted with $S_t$ noise confirmed that the GNLMP algorithm is able to estimate the graph signal robustly with faster convergence.

Acknowledgements This work has been funded by High-end Foreign Expert Talent Introduction Plan under Grant G2021032021L.

Code and Data Availability Statements The code and the data set are available at https://github.com/yan2yi4/GSP_NLMP_paper.

Declarations

Competing Interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Sandryhaila, A., & Moura, J. M. F. (2014). Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. IEEE Signal Processing Magazine, 31, 80–90.
2. Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine, 30*, 83–98.

3. Chung, F. K. (1997). Spectral graph theory. *CBMS Regional Conference Series in Mathematics, 92*.

4. Dong, X., Thanou, D., Toni, L., Bronstein, M., & Frossard, P. (2020). Graph signal processing for machine learning: A review and new perspectives. *IEEE Signal Processing Magazine, 37*, 117–127.

5. Hammond, D. K., Vandergheynst, P., & Gribonval, R. (2011). Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis, 30*, 129–150.

6. Ortega, A., Frossard, P., Kovačević, J., Moura, J. M. F., & Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE, 106*, 808–828.

7. Huang, W., Goldsberry, L., Wymbs, N. F., Grafton, S. T., Bassett, D. S., & Ribeiro, A. (2016). Graph frequency analysis of brain signals. *IEEE Journal of Selected Topics in Signal Processing, 10*, 1189–1203.

8. Di Lorenzo, P., Barbarossa, S., Banelli, P., & Sardellitti, S. (2016). Adaptive least mean squares estimation of graph signals. *IEEE Transactions on Signal and Information Processing, 2*, 555–568.

9. Spelta, M. J. M., & Martins, W. A. (2020). Normalized LMS algorithm and data-selective strategies for adaptive graph signal estimation. *Signal Processing, 167*.

10. Colonnese, S., Petti, M., Farina, L., Scarano, G., & Cuomo, F. (2021). Protein-protein interaction prediction via graph signal processing. *IEEE Access, 9*, 142681–142692.

11. Mohan, D. M., Asif, M. T., Mitrovic, N., Daewuls, J., & Jaillet, P. (2014). Wavelets on graphs with application to transportation networks. In *IEEE Conference on Intelligent Transportation Systems* (pp. 1707–1712).

12. Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS* (pp. 3844–3852).

13. Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *ICLR*.

14. Di Lorenzo, P., Banelli, P., Barbarossa, S., & Sardellitti, S. (2017). Distributed adaptive learning of graph signals. *IEEE Transaction Signal Processing, 65*, 4193–4208.

15. Kuruoglu, E. E., Rayner, P. J., & Fitzgerald, W. J. (1998). Least lp-norm impulsive noise cancellation with polynomial filters. *Signal Processing, 69*, 1–14.

16. Arce, G. (2005). Nonlinear signal processing: A statistical approach. Wiley.

17. Chen, Y., So, H., & Kuruoglu, E. E. (2016). Variance analysis of unbiased least lp-norm estimator in non-Gaussian noise. *Signal Processing, 122*, 190–203.

18. Banerjee, S., & Agrawal, M. (2013). Underwater acoustic communication in the presence of heavy-tailed impulsive noise with bi-parameter Cauchy-Gaussian mixture model. In *SYMPOL* (pp. 1655–1661).

19. Karakus, O., Kuruoglu, E. E., & Altinkaya, M. (2020, November). Modelling impulsive noise in indoor powerline communication systems. *Signal, Image and Video Processing, 14*, 1655–1661.

20. Nguyen, N. H., Doan, K., & Kuruoglu, E. E. (2019). An iteratively reweighted instrumental-variable estimator for robust 3-D AOA localization in impulsive noise. *IEEE Transactions Signal Processing, 67*, 4795–4808.

21. Herranz, D., Kuruoglu, E. E., & Toffolatti, L. (2004, September). An $a$-stable approach to the study of the (P)D distribution of unresolved point sources in CMB sky maps. A&A, 424, 1081–1096.

22. Kuruoglu, E. (2003). Analytical representation for positive alpha-stable densities. In *ICASSP* (pp. VI–729). volume 6.

23. Albu, F., Caciula, I., Li, Y., & Wang, Y. (2017). The lp-norm proportionate normalized least mean square algorithm for active noise control. In *ICSTCC* (pp. 396–400).

24. Nikias, C. L., & Shao, M. (1995). Signal processing with alpha-stable distributions and applications. *Wiley-Interscience*.

25. Shin, J., Kim, J., Kim, T.-K., & Yoo, J. (2021). Lp-norm-like affine projection sign algorithm for sparse system to ensure robustness against impulsive noise. *Symmetry, 13*.

26. Nguyen, N. H., Doan, K., & Wang, W. (2020). Adaptive estimation and sparse sampling for graph signals in alpha-stable noise. *Digital Signal Processing, 105*, 102782.

27. Shi, W., Li, Y., & Wang, Y. (2020). Noise-free maximum correntropy criterion algorithm in non-gaussian environment. *IEEE Transactions on Circuits and Systems II: Express Briefs, 68*, 1547–1551.

28. Radhika, S., Albu, F., & Chandrasekar, A. (2021). Steady state mean square analysis of standard maximum versoria criterion based adaptive algorithm. *IEEE Transactions on Circuits and Systems II: Express Briefs, 68*, 1547–1551.

29. Kutlay, M., & Gagula-Palalic, S. (2016, March). Application of machine learning in healthcare: Analysis on MHEALTH dataset. *Southeast Europe Journal of Soft Computing, 4*, 46–51.

30. Micheloyannis, S., Pachou, E., Stam, C. J., Vourkas, M., Erimaki, S., & Tsirka, V. (2006). Using graph theoretical analysis of multi channel EEG to evaluate the neural efficiency hypothesis. *Neuroscience Letters, 402*, 273–277.

31. Wang, C., Zhu, Y., Zang, T., Liu, H., & Yu, J. (2021). Modeling inter-station relationships with attentive temporal graph convolutional network for air quality prediction. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (pp. 616–634). New York, NY, USA: Association for Computing Machinery.

32. Yan, Y., Adel, R., & Kuruoglu, E. E. (2021). Adaptive normalized LMP estimation for graph signal processing. *IEEE MLSP*.

33. Di Lorenzo, P., Banelli, P., Isufi, E., Barbarossa, S., & Leus, G. (2018). Adaptive graph signal processing: Algorithms and optimal sampling strategies. *IEEE Transaction Signal Processing, 66*, 3584–3598.

34. Arkan, O., Belge, M., Cetin, A. E., & Erzin, E. (1995). Adaptive filtering approaches for non-Gaussian stable processes. In *ICASSP* (Vol. 2, pp. 1400–1403).

35. National Oceanic and Atmospheric Administration. U. S. hourly climate normals. https://www.ncei.noaa.gov/data/normals-hourly/2006-2020/doc/

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.