Normalized LMS for Adaptive Graph Signal Estimation on Genetic Data

Yi Yan, and Radwa Adel

Abstract

In this project, we explored the effectiveness of the GSP NLMS algorithm on modeling time vertex graph genetic data. Analyzing the change in gene expression patterns over time reveals many features of the mechanistic drivers characterizing cellular responses. However, due to the cost of microarray experiments and the limited availability of biological material, most microarray time-series experiments are short, and not many experiments on modeling such data were done. To simulate missing genes in noisy data, the NLMS algorithm is being compared with the LMS algorithm and the RLS algorithm on modeling band-limited graph signal with a reduced number of nodes. The performance of NLMS on steady-state graph signal and time vertex graph signal is being evaluated using MSE, MSD, run time, and convergence speed. In both scenarios, the NLMS algorithm can converge faster than the LMS algorithm and have a lower run time than the RLS algorithm under similar MSD and MSE performance.

1. Introduction

The cell is the basic unit of all living tissue, each cell has a certain function, and this function is determined by the gene expression process. Gene expression is the process done by cells to build protein, and proteins dictate cell function. Recently, Microarray technology has enabled the interrogation of gene expression data in a global and parallel fashion; Microarray technology has become the most popular platform in the era of systems biology [1]. The measurement of gene expression is an important key element in the study of life sciences. Analyzing the level of a gene expression in a cell can provide valuable information in terms of identifying viral infection of a cell (viral protein expression), determining the susceptibility of an individual to cancer (oncogene expression), and finding if a bacterium is resistant to penicillin (beta-lactamase expression). Also, analyzing the gene expression levels over time can infer more information about the mechanism of the biological process [2]. The authors in [3] showed that the oxygen-dependent genes are not part of the previously described environmental stress response (ESR) consisting of genes that respond to diverse types of stress. However, due to the cost of microarray experiments and the limited availability of biological material, most microarray time-series experiments are short (3-8 time points). Besides, the nature of the biological data is noisy and irregular, which makes many computational models fall short of modeling biological data. Previous research [4] tried to build a model that can uncover the hidden patterns of the regularity networks and the transcription process using short time-series gene expression data.

However, these models still unable to infer many of the biological information that governs the cellular system. Thus, in this project, we applied a model that can reconstruct the short time series expression data into a longer time series, which can

enable further traditional time series methods to analyze the gene expression data more effectively and has better results than analyzing short time-series data. This will not only accelerate the biological research process but also the drug discovery and development process because the analysis of the gene expression level can indicate whether the cell is influenced by the targeted drug or not.

The motivation behind defining the gene interaction network on a graph is that the graph structure exploits the dependencies among genes in an interaction network, which has significant importance in uncovering hidden patterns in the complex biological processes. Graph signal processing (GSP) tools such as frequency/spectral domain representation, bandlimited properties, sampling, and reconstruction strategies on graphs have shown promising results in processing irregular data. Recently, applying adaptive filtering to graph signal estimation problems has enabled time-sequential reconstruction and time-varying graph signal tracking in noisy environments [5]. However, many of these algorithms have some drawbacks such as slow runtime or large computational complexity. We explored the effectiveness of the accurate and efficient GSP normalized least mean square (NLMS) algorithm [5] on genetic data [6]; the NLMS algorithm has shown promising results in time-sequential reconstruction of climate data. To the best of our knowledge, NLMS has not been used to model gene expression.

2. Background Material

2.1 Gene Expression

The cell is the basic unit of all living tissue, in most human cells there is a structure called the nucleus. It contains the genome, in humans, the genome is split between 23 pairs of chromosomes, each chromosome contains a long strand of DNA, tightly packaged around proteins called histones. Within the DNA there are sections called genes, these genes contain the instructions for making proteins. When a gene is switched on an enzyme called RNA polymerase attaches to the start of the gene. It moves along the DNA making a strand of messenger RNA (mRNA) out of free basis in the nucleus, the DNA code determines the order in which the free basis is added to the messenger RNA and this process is called transcription. Before the messenger RNA can be used as a template for the production of proteins, it needs to be processed, this involves removing and adding sections of RNA, the messenger RNA then moves out of the nucleus into the cytoplasm. Protein factories in the cytoplasm called ribosomes bind to the messenger RNA, the ribosome reads the code in the messenger RNA to produce a chain made up of amino acids, there are 20 different types of amino acids.

Transfer RNA molecules carry the amino acids to the ribosome, the messenger RNA, and is read three bases at a time. As each triplet is read a transfer RNA delivers the corresponding amino acid, this is added to a growing chain of amino acids. Once the last amino acid has been added, the chain folds into a complex 3D shape, to form the protein.

2.2 Gene Expression Level

Measurement of expression is done by detecting the final gene product (for many genes, this is the protein); however, it is often easier to detect one of the precursors, typically mRNA, and to infer gene-expression levels from these measurements.

2.3 Graph Theory

A graph is formulated as G = {V, E} where V is the set of N nodes, and E is the set of edges. There are two types of graphs: directed and undirected. A directed graph is when the edge between two points include a direction indicator; undirected graphs have no direction indications. In the undirected unweighted graph, which is the type of graph used in this project, the adjacency matrix is a mathematical representation of a graph, denoted as **A**, which is an N by N matrix, with $\mathbf{A}_{ij} = 1$ if (v_i, v_j) are connected, and $\mathbf{A}_{ij} = 0$ if (v_i, v_j) are not connected. The degree matrix **D** is a diagonal matrix indicating the number of edges attached to each node.

2.4 Graph Construction Methods

There are several popular constructions to transform a given set $x_1, \ldots, x_n$ of data points with pairwise similarities s_{ij} or pairwise distances d_{ij} into a graph. When constructing similarity graphs the goal is to model the local neighborhood relationships between the data points, such as the ε-neighborhood graph, k-nearest neighbor graphs, and the fully connected graph [4].

2.5 Adaptive Signal Processing

In a supervised adaptive filter system one has a desired signal $d[k] \in R$, an input $x[k] \in R^N$, and the parameter vector $\hat{\vec{h}}_0[k] \in R^N$, which is an estimate of the possibly time-varying unknown system parameters $\vec{h}_w[k] \in R^N$. From this simple configuration an instantaneous error signal is defined as

$$e[k] = d[k] - \mathbf{x}^T[k]\hat{\boldsymbol{h}}_o[k].$$

2.6 Graph Signal Processing

After extracting the adjacency matrix **A** and degree matrix **D**, the Graph Laplacian matrix is calculated by subtracting the adjacency matrix from the degree matrix:
L=D-A.

To transform the data from the time domain to the spectral domain, use the graph Fourier transform (GFT) of the graph signal (GS). In this project it is the gene expression level of each node. A GS is defined as GS $x \in R^N$. The GFT of the GS is its projection onto a set of orthonormal vectors, for which use the orthonormal eigenvectors of the **L** matrix, the GFT is giving as the formula below:

$$\boldsymbol{s} = \mathbf{U}^T\boldsymbol{x}.$$

To transform the GS from the spectral domain, after we estimate it, we use the Inverse graph Fourier transform as in the formula below:

$$\boldsymbol{x} = \mathbf{U}\boldsymbol{s}.$$

A graph signal is bandlimited or spectrally sparse when its frequency-domain representation **S** has zero entries, we changed our data to be bandlimited by the strategy mentioned in [5]. We applied the sampling technique as in [5] to ensure a reduced number of nodes used in our model.

2.7 The GSP LMS Algorithm

The GSP LMS algorithm [7] is a popular adaptive GSP algorithm due to its simplicity. The main idea of the GSP LMS algorithm is to apply the least mean square method onto graph signal using spectral domain techniques. It is a convex optimization problem defined as the following:

$$\min_{\tilde{\boldsymbol{s}}_{\mathcal{F}}} \mathbb{E}\left[\|\mathbf{D}_{\mathcal{S}[k]}(\boldsymbol{x}_w[k] - \mathbf{U}_{\mathcal{F}}\tilde{\boldsymbol{s}}_{\mathcal{F}})\|_2^2 \right].$$

The $\vec{x}_w[k]$ is the noisy reference obtained by adding gaussian noise to the original data $\vec{x}_0[k]$. Ds is the sampling matrix. The prediction in the spatial domain is done through iGFT of the prediction $\vec{s}_f[k]$ in the spectral domain. The update function in spatial domain is obtained in [3] and is shown as the following:

$$\hat{\mathbf{x}}_o[k+1] = \hat{\mathbf{x}}_o[k] + \mu_L \mathbf{U}_{\mathcal{F}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k].$$

Here $\hat{x}_0[k]$ is our prediction in the spatial domain. The convergence factor μ_L is a constant $0 < \mu_L < 2$ that acts as step size to balance the convergence behaviour and reducing the steady-state error of the algorithm.

2.8 The GSP RLS Algorithm

The GSP RLS algorithm [8] is changing the objective function of the LMS algorithm form LMS to weight Least-Squares:

$$\min_{\tilde{\mathbf{s}}_{\mathcal{F}}} \sum_{l=1}^{k} \beta_R^{k-l} \|\mathbf{D}_{\mathcal{S}[l]}(\mathbf{x}_w[l] - \mathbf{U}_{\mathcal{F}}\tilde{\mathbf{s}}_{\mathcal{F}})\|^2_{\mathbf{C}_w^{-1}[k]} + \beta_R^k \|\tilde{\mathbf{s}}_{\mathcal{F}}\|^2_{\mathbf{\Pi}}.$$

The forgetting factor β_R is in the range $0\ \beta R \leq 1$ and has similar functionality as μ_L in the LMS algorithm. We will be using the RLS algorithm to compare the performance of the NLMS algorithm. For detailed analysis of the RLS algorithm, please refer to [8].

3. Methodology

3.1 Approach Overview

The data we are using is synthetic data generated by GeneNetWeaver [6]. The data comes as a GS with steady-state GS and time series GS and is discussed in detail in the data section. We reconstruct the data into a time vertex graph and compute the Laplacian matrix for GSP. Then we transform the data into its spectral domain by Graph Fourier Transform. To obtain a spare representation of the GS, we generate a band-limited GS through spectral-domain filtering. Then spectral-domain sampling strategy used in [5] is applied to sample several nodes in the graph to use in our model, and the unsampled nodes are dropped during the modeling. After the previous preprocessing steps, we iteratively apply the GSP NLMS algorithm to model the GS in the spectral domain.

Lastly, the spectral domain results are converted back to the special domain through inverse graph Fourier Transform. To compare the performance of the GSP NLMS algorithm, the MSE, MSD, runtime, and convergence time for sampled and unsampled data are calculated for evaluation. Comparison is being made among other GSP algorithms, namely the GSP LMS algorithm [7] and the GSP RLS [8] algorithm. We average the results of the models across 200 runs to eliminate outliers.

3.2 The GSP NLMS Algorithm

The GSP NLMS algorithm was first introduced by [5] and applied to temperature data. In [5], NLMS was used for estimating time-series temperature data fitted to a separately generated graph that is not part of the data. In this project, we propose to apply the GSP NLMS algorithm on genetic data that is inherently defined on a graph. Intuitively, the GSP NLMS algorithm is the following constrained convex problem of minimizing the distance between the current and the updated estimate in the spectral domain:

$$\begin{aligned} \underset{\hat{\mathbf{s}}_{\mathcal{F}}[k+1]}{\text{minimize}} \quad & \|\hat{\mathbf{s}}_{\mathcal{F}}[k+1] - \hat{\mathbf{s}}_{\mathcal{F}}[k]\|_2^2 \\ \text{subject to} \quad & \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{S}[k]} (\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}} \hat{\mathbf{s}}_{\mathcal{F}}[k+1]) = \mathbf{0} \end{aligned}.$$

This is a convex optimization problem and the analytical solution is given using gradient decent [5]. Similar to the GSP LMS algorithm in [7], the GSP NLMS algorithm is aimed to minimize the squared error which is defined as in the following equation:

$$\vec{e}[k] = \vec{D}_{\mathcal{S}[k]} (\vec{x}_w[k] - \hat{\vec{x}}_o[k]).$$

Here $\vec{x}_w[k]$ is the noisy reference and $\hat{\vec{x}}_0[k]$ is our prediction in the spatial domain. $\vec{x}_w[k]$ is obtained by adding gaussian noise to the original signal $\vec{x}_0[k]$. The sapling matrix D_s makes sure we obtain the reduced number of nodes used in our model. Based on the derivation of the iterative solution of NLMS, given in [5] and using GFT to transform the error into the spectral domain, the next step prediction results in the spectral domain is

$$\hat{\mathbf{s}}_{\mathcal{F}}[k+1] = \hat{\mathbf{s}}_{\mathcal{F}}[k] + (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{S}[k]} \mathbf{U}_{\mathcal{F}})^{-1} \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{e}[k].$$

Applying inverse GFT, the special domain next step prediction results in the spectral domain is

$$\hat{\mathbf{x}}_o[k+1] = \hat{\mathbf{x}}_o[k] + \mu_N \mathbf{U}_{\mathcal{F}} (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{S}[k]} \mathbf{U}_{\mathcal{F}})^{-1} \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{e}[k].$$

The convergence factor μ_N is a constant $0 < \mu_N < 2$ that acts as step size similar to μ_L in the GSP LMS algorithm.

Algorithm 1 NLMS estimation of graph signals.

$k \leftarrow 0$
while (true) **do**
  $\vec{e}[k] = \vec{D}_{\mathcal{S}[k]}(\vec{x}_w[k] - \hat{\vec{x}}_o[k])$
  $\hat{\vec{x}}_o[k+1] = \hat{\vec{x}}_o[k] + \mu_N \vec{U}_{\mathcal{F}}(\vec{U}_{\mathcal{F}}^T \vec{D}_{\mathcal{S}[k]} \vec{U}_{\mathcal{F}})^{-1} \vec{U}_{\mathcal{F}}^T \vec{e}[k]$
  $k \leftarrow k+1$
end

4. Data description

The Synthetic data is generated by GeneNetWeaver [6], which is given us as a GS and each gene is represented as a node. A subnetwork is extracted from the complete dataset and the structure is from known transcriptional networks (Escherichia coli, Saccharomyces cerevisiae, etc.) as shown in Figure 1. GeneNetWeaver [6] also generates the time series GS that comes along with the graph.

To generate the data, GeneNetWeaver [6] simulated the generated in silico regulatory networks to produce synthetic gene expression datasets. In other words, each measurement point is generated by applying multifactorial perturbations to the network. One may think of each experiment as a gene expression profile from a different patient, For example, they simulated multifactorial perturbations by slightly increasing or decreasing the basal activation of all genes of the network simultaneously by different random amounts.

Figure 2 represents the gene expression level as a graph signal defined on the gene network, each plot represents different measurement point, each color shade represents a different quantity of the expression level as shown on the color column.

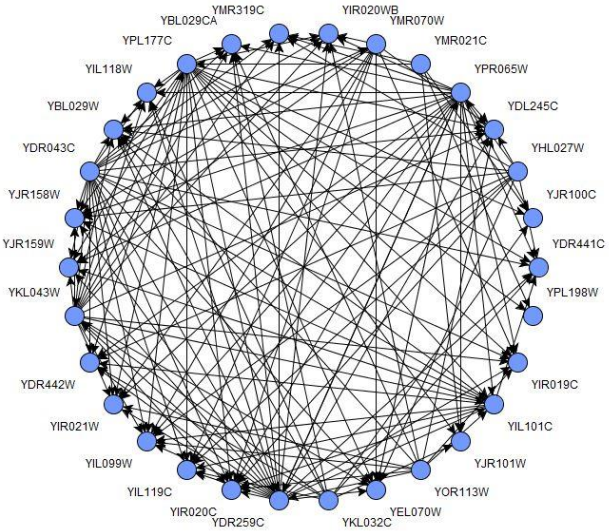

Figure 1. Graph network structure of 30 different yeast genes generated by GNW

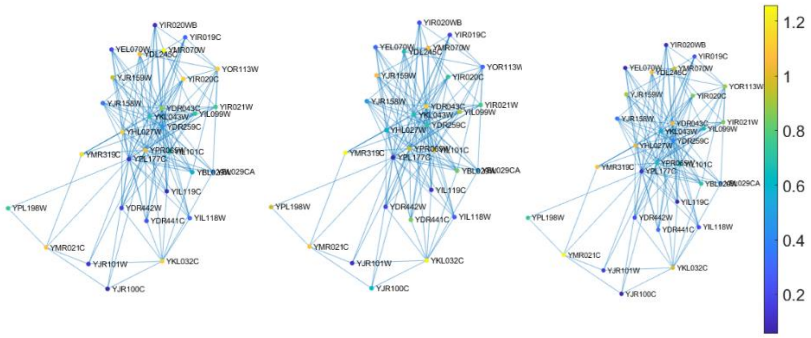

Figure 2. Graph signal representation of 3 time points at t =0, t = 125, and t = 251 from the time series generated by GNW.

5. Experimental study

5.1 Steady State GSP modelling

For comparing the performance of the GSP LMS, GSP RLS, and GSP NLMS algorithms on modelling steady state graph signal, we used a steady state GS of 30 nodes generated by GeneNetWeaver [6]. For this project, $|S| = 29$, which means only sampling 29 nodes in our modelling, and $|F| = 28$ for band-limited GS. The step size is $\mu_L = \mu_N = 1$ and $\beta_R = 0.1$ based on grid search of best performing step size in MSD measure. The MSD and MSE of modelling steady state GS is shown in Figure 3 and Figure 4. Table 1 summarizes the run time of modeling a steady-state GS for 50 iterations. To eliminate outliers, the experiment is being repeated 200 times, and the averaged reconstruction of the GS is being used to evaluate the performance.

Table 1. The time in seconds for the algorithms to execute 50 iterations, averaged across 200 runs.

LMS	RLS	NLMS
46.2795e-006	1.5220e-003	45.6065e-006

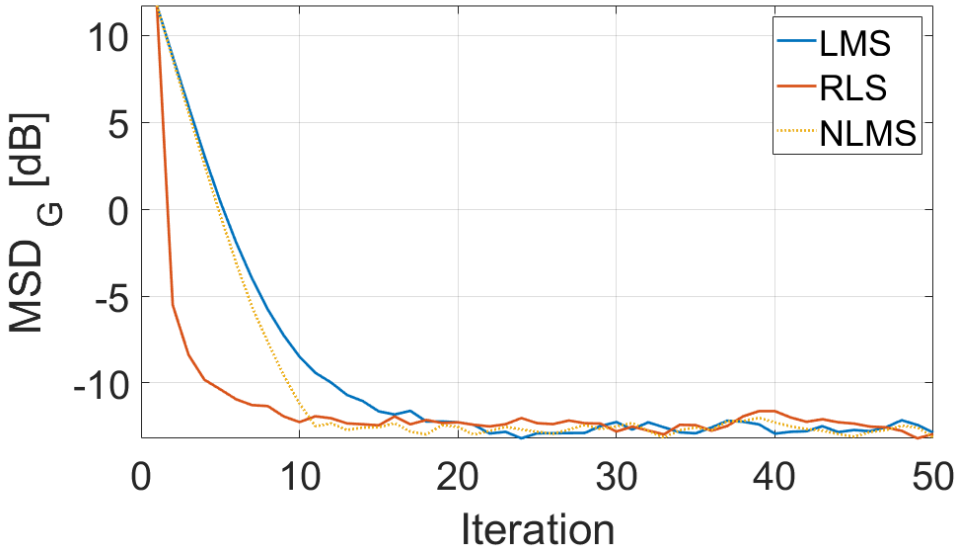

Figure 3. MSD of modeling steady state GS.

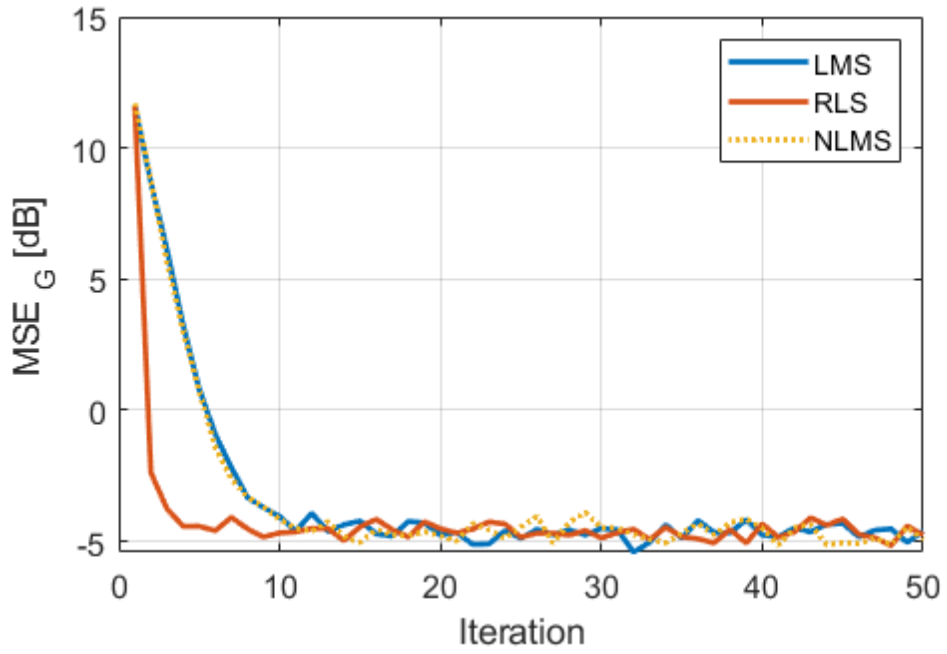

Figure 4. MSE of modeling steady state GS.

5.2 Time Series GSP modelling

To model time series GS we used a GS of 30 nodes and 251 time steps generated by GeneNetWeaver[6]. Again, the GSP NLMS algorithms is being compared with the GSP LMS and GSP RLS in MSD, MSE, and run time. The hyperparameter set up follows the previous section where $|S| = 29$, $|F| = 28$, $\mu_L = \mu_N = 1$, and $\beta_R = 0.1$. Figure 5 and Figure 6 gives a visualization of adaptive estimates across time on a sampled node and on an unsampled node. From Figure 5, we can see that all three algorithms

model the original data well. To see the detailed comparison of the three models, we calculate the MSD at each time step for each node, and the summed MSD across the 30 nodes at each time step is shown in Figure 7 and Figure 8. Also, the run time is being calculated to compare the performance of three algorithm in Table 2. Again, the experiment is being repeated 200 times and the model is being averaged to eliminate outliers. Here we are dealing with time series data, so each iteration is a single time step, so the algorithms runs 251 iterations to predict 251 time steps.

Table 2. The time in seconds for the algorithms to execute, averaged across 200 runs.

LMS	RLS	NLMS
227.8310e-006	5.8737e-003	213.8850e-006

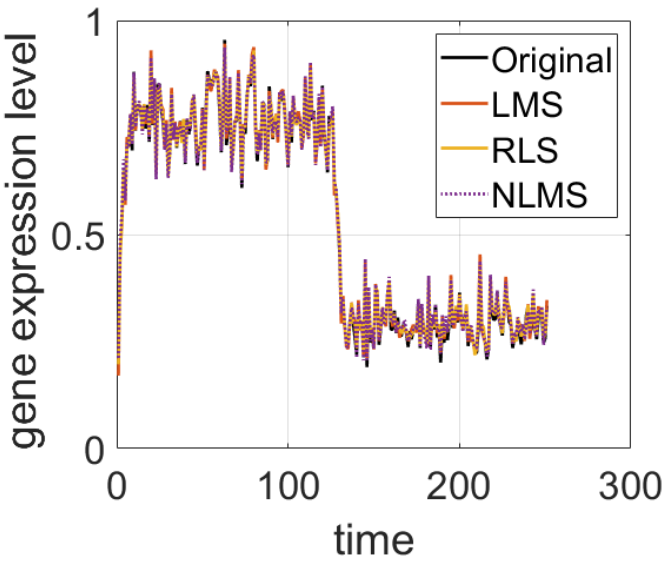

Figure 5. Adaptive estimates across time on a sampled node.

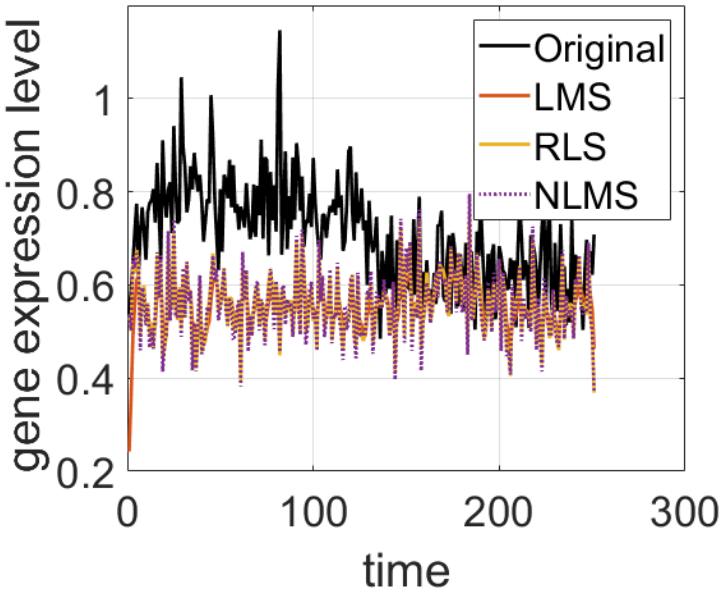

Figure 6. Adaptive estimates across time on an unsampled node.

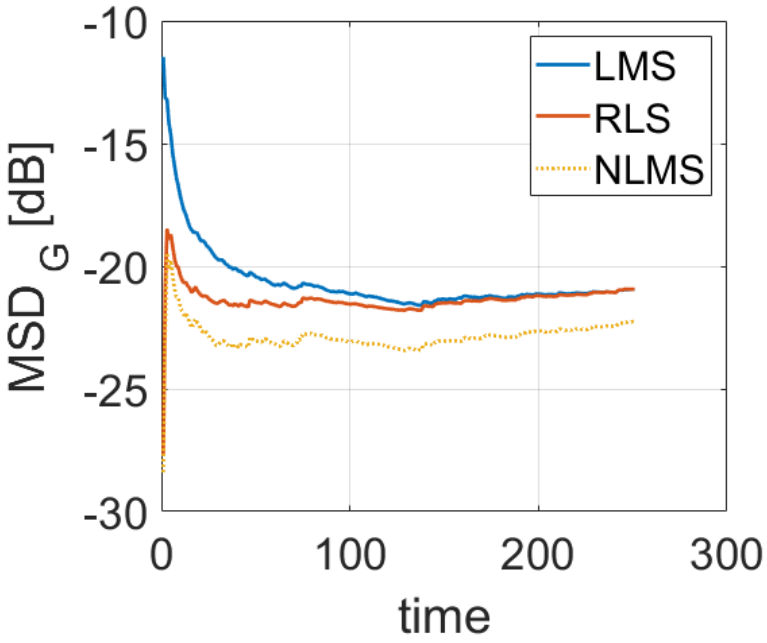

Figure 7. The sum of the MSD on the 29 sampled nodes at each iteration.

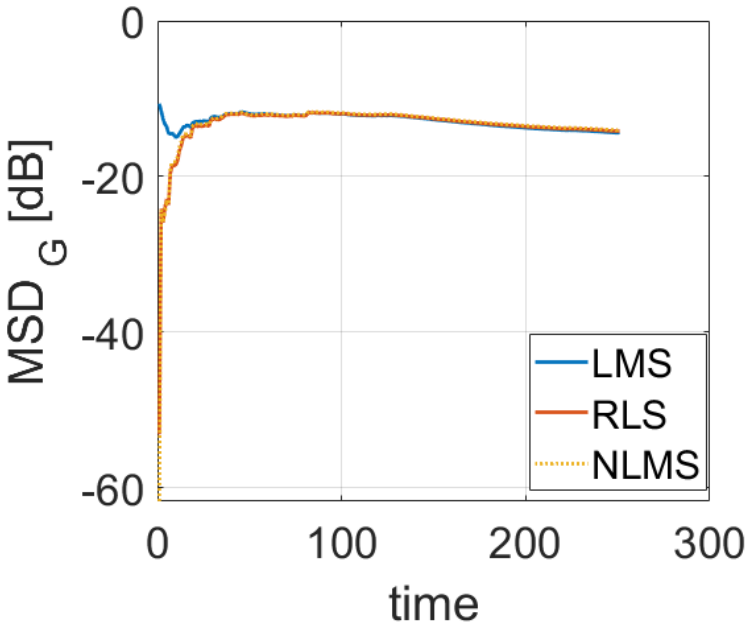

Figure 8. The MSD on the unsampled node at each iteration.

6 Discussions

6.1 Steady State GSP modelling

From Figure 3, we can see that the LMS algorithm converges at iteration 24, the RLS algorithm converges at iteration 10, and the NLMS algorithm converges at iteration 11. The fast convergence behavior of the NLMS comparing to LMS is not obvious for the MSE because of the way the MSE and MSD are defined in the project. In this project, MSD is the deviation from the actual signal, where MSE is the error between our prediction and the noisy signal. In principle the NLMS is a normalized version of LMS, so the error mechanism of the two algorithms are similar. In this project the noise behavior is not the same as the actual signal, so adding the noise and measuring MSE in this way does not reflect how the algorithms model the actual GS. Because the 3 algorithms have different computational complexities, comparing the number of iterations does not indicate the actual run time, so the runtime is measured as well. From the comparison of run time in Table 1, even though RLS converges in less number of iteration, due to the higher computational complexity of RLS comparing to the LMS and the NLMS, the NLMS algorithm converges the fastest among the three algorithms.

6.2 Time Series GSP modelling

From the MSD plots in Figure 7 and Figure 8, the GSP NLMS algorithm has the lowest MSD on sampled data among all three algorithms. The three algorithms performed similarly on the unsampled nodes; this is due to the small size of our input graph. As the size of the graph increase, the intuition is that more nodes and edges of the model will let the sampled nodes include more information about the unsampled nodes in the spectral domain. So, combining the good performance of the NLMS

algorithm in the sampled data, the lower MSD of the NLMS algorithm on unsampled data should be more obvious than the LMS and the RLS when we use a larger model size. This study will be conducted in the follow-up work.

6.3 Potential Improvements and Future Work

The modeling of unsampled nodes could be seen as underlying gene expression process such as inferring causality from the temporal response pattern modeling. We could use the traditional time-series methods for further analysis of the result. We could also explore the difference between applying these traditional time-series methods on the original data and the predicted data, to see if results from applying the traditional time-series methods on the original data will infer the same hidden patterns of the biological processes when applying these methods on the predicted values.

The original graph of the regularity network was directed. However, for the complexity of the GFT for directed graphs in this project, we removed the directions from the graph. This might have affected the results of the applied GSP NLMS algorithm. Thus, to develop this model we can use directed GFT in the follow-up work, to examine whether the directions of the graph will improve the result of the applied method or not. Another approach is to change the square of the NLMS algorithm into a fractional number $1<p<2$, and make this into a GSP NLMP algorithm. Also the added noise is gaussian, it would be more interesting if we model the noise using non gaussian distribution such as alpha stable distribution.

7 Conclusions

The GSP NLMS has faster convergence speed and run speed than the GSP LMS with similar performance at MSE and MSD. The GSP NLMS has faster computation time than the GSP RLS with similar performance at MSE and MSD. The model can be further used for other irregular data such as social networks data and traffic data. The model performance improves as the graph size increase and the graph get more connected. The model can be further improved by considering the direction of the edges representing genetic interactions.

References:

[1] Panda S, Sato TK, Hampton GM, Hogenesch JB: An array of insights: application of DNA chip technology in the study of cell biology. Trends in cell biology. 2003, 13 (3): 151-156.

[2] Aguayo-Orozco Alejandro, B. F. (2018). Analysis of Time-Series Gene Expression Data to Explore Mechanisms of Chemical-Induced Hepatic Steatosis Toxicity . Frontiers in Genetics , 396.

[3] Nasrine Bendjilali, Samuel MacLeon, Gurmannat Kalra, Stephen D. Willis, A. K. M. Nawshad Hossian, Erica Avery, Olivia Wojtowicz and View ORCID ProfileMark J. Hickman, G3: GENES, GENOMES, GENETICS January 1, 2017 vol. 7 no. 1 221-231; https://doi.org/10.1534/g3.116.034991

[4] Wang, X., Wu, M., Li, Z. et al. Short time-series microarray analysis: Methods and challenges. BMC Syst Biol 2, 58 (2008). https://doi.org/10.1186/1752-0509-2-58

[5] Marcelo Jorge Mendes Spelta, W. A. (2020). Normalized LMS algorithm and data-selective strategies for adaptive graph signal estimation. Signal Processing, 0165-1684.

[6] Schaffter T, M. D. (2011). GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. Bioinformatics (Oxford, England), 2263-2270.

[7] P.D. Lorenzo, S. Barbarossa, P. Banelli, S. Sardellitti, Adaptive least mean squares estimation of graph signals, IEEE Trans. Signal Inf. Process. Netw. 2 (4) (2016) 555–568, doi:10.1109/TSIPN.2016.2613687.

[8] P.D. Lorenzo, E. Isufi, P. Banelli, S. Barbarossa, G. Leus, Distributed recursive least squares strategies for adaptive reconstruction of graph signals, in: 2017 25th European Signal Processing Conference (EUSIPCO), 2017, pp. 2289–2293, doi:10.23919/EUSIPCO.2017.8081618.