Fourier Transform for Noise Reduction in Time-Series Data

Tien (V.) Nguyen
Department of Electrical and Computer Engineering
University of Texas at San Antonio
tien.nguyen@utsa.edu
April 23, 2024

***Abstract*—This project explores the application of the Fourier Transform (FT), a widely used technique in applied mathematics, particularly in signal processing and data compression. The Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT) are practical variants of the FT, with FFT being more efficient for large-scale problems. The project addresses the challenges of analyzing measured data and signals, which often contain noise due to unpredictable circumstances and systematic measurement errors. The noise is typically modeled as Gaussian white noise. The project implemented the FT-based amplitude thresholding noise removal framework for denoising. To demonstrate the effectiveness of the framework, we evaluate on a variety of 1D datasets, including synthetic and real-world signals. Based on the performance of the framework, we realized that the implemented framework is useful for reducing noise in data, however, the we should be critical in choosing threshold value to get best performance.**

***Index Terms*—Signal Processing, Noise Reduction, Discrete Fourier Transform, Gaussian white noise**

I. Introduction

The Fourier transform, often abbreviated as FT [1], is a prevalent method in applied mathematics, particularly in signal processing and data compression. Essentially, the FT operates on the principle that certain types of functions can be depicted as a linear combination of trigonometric functions. This transformation process is akin to deriving frequency-domain data from a time-domain signal, which provides a more efficient means for analyzing signals.

The Discrete Fourier Transform (DFT) [2] is a practical and applicable variant of the Fourier Transform (FT) used in real-world scenarios. DFT finds its use in various applications such as data compression [3], solving partial differential equations [4], and dealing with noisy signals [5]. On the other hand, the Fast Fourier Transform (FFT) [6] is a more efficient version of the DFT, particularly for large-scale problems. The key advantage of using FFT is that it reduces the computational cost from $O(N^2)$, as in DFT, to $O(N \log N)$ resulting in an exponential speed-up.

Analyzing measured data and signals comes with its own set of challenges due to unpredictable circumstances and systematic measurement errors [7] [8]. These elements introduce noise into the data, complicating the analysis process and potentially leading to biased or incorrect results. In many applications, it is beneficial to identify the true function to model the underlying process rather than using the noisy measurements directly. In practice, and in accordance with the Central Limit Theorem [9], the noise is usually modeled as Gaussian white noise, a common method for representing the effects of random processes in measured data [10].

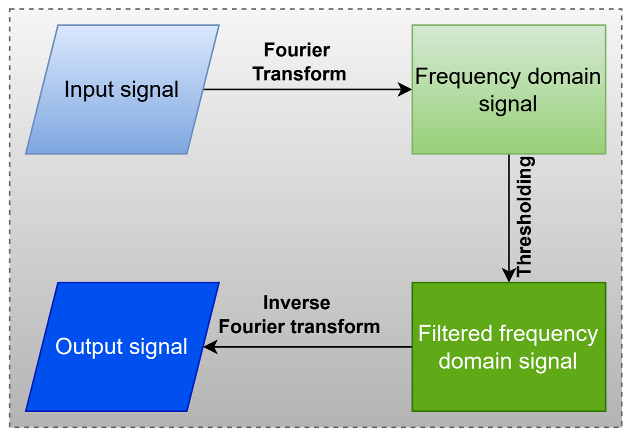


Fig. 1. FT-based noise reduction using amplitude thresholding. First, the input signal with the assumption was noised will be transformed into a frequency domain through the Fourier transform, then we apply some threshold value to cut off the weak frequency components, i.e., set the amplitude of these components to 0, to get the filtered frequency signal. Finally, we transform back to the time domain by using the Inverse Fourier Transform to get the reducing noisy signal.

Typical techniques for reducing noise in one-dimensional data signals encompass adaptive filtering, signal smoothing, and frequency domain noise cancellation [11]. In this project, we are interested in utilizing the Fourier transform method to eliminate noise from the data. Specifically, we aim to assess the FT-based amplitude thresholding noise removal framework, as depicted in figure 1. To accomplish this, the project will:

- Provide a refreshed overview of the noise reduction framework that employs the Fourier transform on Gaussian white noise.
- Develop and demonstrate the framework using Python, with FFT as the central technique.
- Test the framework on a variety of 1D datasets, including synthetic groups such as trigonometric functions and polynomials, as well as real-world groups (IBM stock).

II. BACKGROUNDS

A. Discrete Fourier Transform

Fourier Transforms (FT) break down a function into coefficients that form a linear combination of trigonometric functions. However, in practical scenarios, we often deal with discrete measurements or fine sequences of signals. As a result, we utilize the Discrete Fourier Transform (DFT), which can be expressed as follows:

$$x_\omega[k] = \sum_{n=0}^{N-1} x_t[n] e^{\frac{-2\pi i}{N}kn} \tag{1}$$

where f denotes an arbitrary function dependent on time t and the Fourier transform of f is given as $g(\omega)$ depending on the frequency ω, and $x_\omega[k]$ denotes the k-th element in the discrete Fourier Transform vector and $x_t[n]$ denotes the n-th element in the original sampled signal. The corresponding Inverse Discrete Fourier Transform (IDFT) which is defined as below.

$$x_t[n] = \frac{1}{N} \sum_{k=0}^{N-1} x_\omega[n] e^{\frac{2\pi i}{N}kn} \tag{2}$$

The calculation of DFT and its inverse can be significantly optimized by coupling even and odd functions during computation, a process known as the Fast Fourier Transform (FFT). The computational cost is reduced from $O(N^2)$ to $O(N \log N)$ when transitioning from DFT to FFT. Hence, this project will employ FFT for the forward transformation and its inverse, the Inverse Fast Fourier Transform (IFFT), for the implementation.

B. Gaussian white noise

White noise [12] [13] serves as a theoretical model that characterizes the impact of numerous uncorrelated random variables. A particular type of white noise, known as Gaussian white noise, assumes that the noise follows a normal distribution. In this context, a vector of measurement errors, denoted as $\mathbf{e} = [e_1, \ldots, e_N]^T$, is considered Gaussian white noise if each e_n, adheres to a normal distribution, specifically $\mathcal{N}(0, \sigma^2)$.

Breaking down a time series into its frequency components yields a distribution of spectral density across various frequencies. These spectral density values constitute a power spectrum for the time series. The power spectrum of band-limited Gaussian white noise is as follows:

$$P = \begin{cases} \sigma^2, & |\omega| < B \\ 0, & o.w \end{cases} \tag{3}$$

where B denotes the band-limited frequency and σ^2 is the noise variance. From the theoretical power spectrum, the total power of the noise P_T can be calculated as:

$$P_T = 2B\sigma^2 \tag{4}$$

The distortion caused by noise can be quantified as the total difference between the original noisy function and the modified function after noise reduction.

$$\epsilon_T^2 = ||f(t) - \hat{f}(t)||_2^2 = \int_{-\infty}^{\infty} (f(t) - \hat{f}(t))^2 dt \tag{5}$$

, where $\hat{f}(.)$ denotes estimation function which is attained after deducting noise. Using Plancherel's identity, the difference can be given in the frequency domain as:

$$\epsilon^2 = \int_{-\infty}^{\infty} (g(\omega) - \hat{g}(\omega))^2 d\omega \tag{6}$$

, where $\hat{g}(\omega)$ denotes Fourier transform of the modified function. To estimate the effectiveness of the noise reduction model σ_T^2 can be compared to the theoretical power of the noise if the variance in 4 is known. The optimal method should result in

$$\epsilon_T^2 \approx 2B\sigma^2 \tag{7}$$

In practice, however, the noise variance typically is unknown and the power of the noise can only estimated. An alternative way of computing the effectiveness of the noise reduction is to measure the difference between the modified Fourier transform and the original function.

$$d = ||\hat{f}(t) - f_{true}(t)||_2^2 = \int_{-\infty}^{\infty} (\hat{g}(\omega) - g(\omega))^2 d\omega \tag{8}$$

, where $f_{true}(t)$ denotes the background function without noise. Instead of using integral for continuous values, in practice, we use DFT which calculates by using summations.

C. Denoising approaches

Adaptive noise cancellation locates points where a signal is unpredictable and aims to adjust the value of these points to fit the rest of the data better. Smoothing algorithms reduce noise by smoothing the curve using various techniques. One way this can be accomplished is by calculating the mean value of three consecutive points, thereby reducing noise by partly averaging out the noise.

Alternatively, one could consider smoothing or noise reduction in the frequency domain. As previously mentioned, this can be achieved by the use of Fourier transforms. Noise filtering via FT has seen numerous applications such as canceling out electromagnetic conditions in radar measurements [14], echo suppression in audio processing [15], and 2D noise filter in image processing [16]. Therefore, in this project, we will apply the FT-based amplitude thresholding framework to denoising signals.

III. FT-BASED AMPLITUDE THRESHOLDING NOISE REDUCTION

After performing the DFT to transfer the data from the time domain to the frequency domain, the results are obtained by suitable modification of the Fourier transform coefficients. Because of the power spectrum of white noise, we can choose an amplitude threshold below which all noise components should remain. Using information of the Fourier transform of trigonometric functions, dominant components can be distinguished from the noisy data. After filtering out weaker

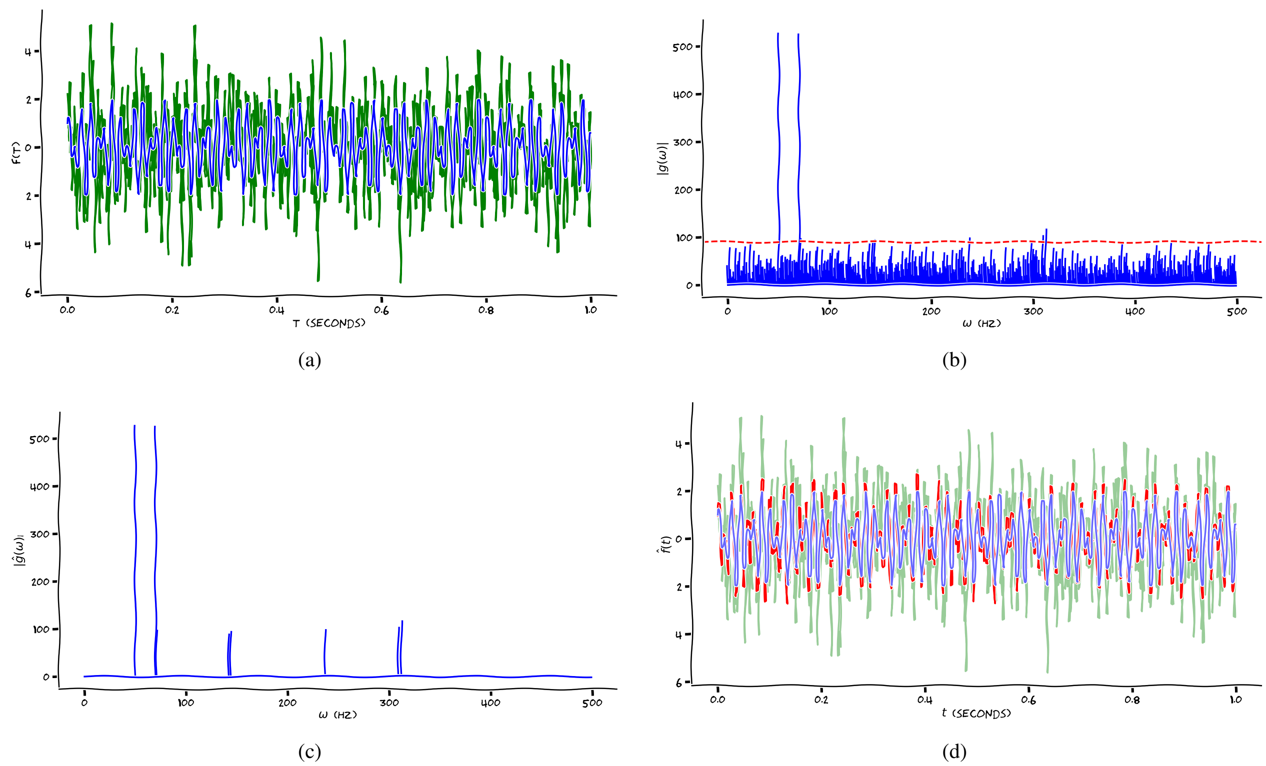


Fig. 2. An example of Fourier transform-based amplitude thresholding framework. First, the acquired signal (green curve) and the no noise signal (blue curve) in figure 2(a) were shown. Next, figure 2(b) shows the action of the Fourier transform which maps the input signal (green curve in figure 2(a) to the frequency domain, and then we remove the weak frequency components (under the dash red line) to get the results as figure 2(c). The denoising signal would be finally obtained by performing the Inverse Fourier transform on the amplitude thresholding frequency signal (figure 2(d). We can see the output signal of the framework (red curve in 1) now is quite close to the cleaning signal, which is expected.

components we can compute the IDFT of the remaining coefficients resulting in a filtered signal in the time domain. The pipeline, in particular, can be described via an example in figure 2.

IV. Numerical studies

A. Datasets

1) Synthetic data: First, to simply, we test these reducing noise techniques above on the artificial dataset. The first synthetic dataset is a linear combination of two trigonometric functions, which is defined as:

$$f_{true} = A_1 \sin(2\pi\omega_1 t) + A_2 \cos(2\pi\omega_2 t) \tag{9}$$

, where ω_1 and ω_2 denote the frequencies of the trigonometric components with A_1 and A_2 being the corresponding amplitudes. The parameter values explicitly are $A_1 = 1/3, A_2 = 1/2, \omega_1 = 19, \omega_2 = 19$. For the adding white noise, we set its variance $\sigma^2 = 1$. Data points were generated using 1000 linearly spaced points with $t \in [0, 1]$.

Next, we test the methods on a polynomial given by this formula:

$$f_{true}(t) = \sum_{i=0}^{p} a_i t^i \tag{10}$$

where $a_i \in \mathbb{R}$ are parameters, and x_i are variables. More particularly, we set these values as $p = 5, a_0 = 1, a_1 = 0, a_2 = 12, a_3 = -60, a_4 = 120, a_5 = -70$. The number of samples would be generated as the previous synthetic dataset.

2) Real-world data: As a practical case, we apply the methods discussed above to IBM stock price data[1]. There are several fields in the data including *time, open, high, low, close, volume*. For this work, we only consider two fields: *time*, and *close* representing the acquired time and the close price of stock respectively. The data originally were collected every minute, and from 04/06/2020 to 03/25/2022. Instead of using the entire data, we consider 2000 samples which sampled uniformly from 03/01/2022 (figure 3).

B. Evaluation metrics

To evaluate these reducing noise methods on synthetic data, we mean square error between denoising signal and original signal without noise, d. The formula to calculate d is supposedly given as (8), however, due to the discrete property, it would be stated under summations as:

$$d = \frac{1}{N} \sum_{t=0}^{N-1} (\hat{f}(t) - f_{true}(t))^2) \tag{11}$$

On the IBM stock dataset, we use the Signal to Noise Ratio (SNR) [17] measurement to report how well the noise

[1] https://www.kaggle.com/datasets/bhanuprasanna527/stock-market-prediction

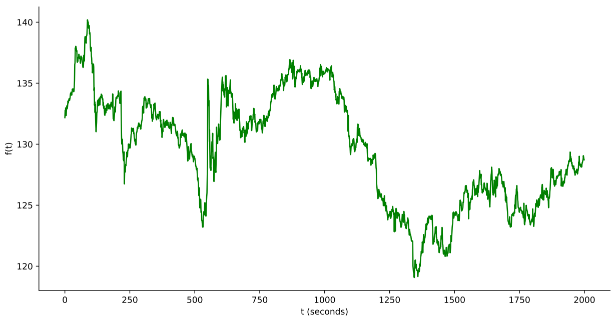


Fig. 3. Visualization of IBM stock market dataset using in the experiment. The time $t = 0$ is equivalent to 1/3/2022 at 4:13 and the time $t = 2000$ is equivalent to 03/25/2022 at 19:19.

reduction framework did. However, due to a lack of knowledge about the true signal, we use the approximation version, which is the reciprocal of square of mean of clean signal and variance of removed noise.

$$\text{SRN} = \frac{\mu^2_{\text{clean}}}{\sigma^2_{\text{removal}}} \tag{12}$$

, with $\mu_{clean} = \frac{1}{N}\sum_{t=0}^{N-1}\hat{f}(t)$ is the expected value after cleaning operation, and $\sigma^2_{removal} = \frac{1}{R}\sum_{r=1}^{R}(k_r - \mu_{removal})^2$ is the variance of removal components in frequency domain with R is the total number of removed components, k_r is the r-th component in set of removed components, $\mu^2_{removal}$ is the mean of removal frequencies.

C. Results

1) Signal reconstruction: Figure 4 shows us the effective of threshold on different types of reconstructed signals. Two threshold 0.2 and 0.4 were applied for trigonometric and polynomial signals, while $1e^{-3}$ and $1e^{-2}$ were used in IBM dataset. Subfigure 4(a) shows that the threshold value 0.4 gave better result when the reconstructed signal is really good in term of approximation of true signal (without noise). On the other hand, when we switch on the polynomial type, the results of using the lower threshold, 0.2 now is better. In particular, the cyan curve in subfigure 4(b) is a straight line which is equal to the mean of signal. This is caused by the removing too many components when doing the cut-off frequencies, or by the high threshold, 0.4.

On the figure 4(c), the lower threshold $1e^{-3}$ seems to be better due to it is quite close the the signal, but this is uncertain since we do not have the knowledge about the true signal (the blue curve now is unavailable). Therefore, in this case, we need to use a specific evaluation metric to evaluate how every model performs successfully, especially using signal to noise ratio.

These results above suggest us that need to find the optimal threshold for different types of data to achieve the best configuration base on the RMS metric. This also give us the sense of the usefulness of using SNR metric in case of lack of knowledge about true function, which is the practical problem we will face in real world.

2) Optimal threshold: The figure 5 demonstrated performance of FT-based denoising framework with different threshold ratios. On two synthetic datasets, we use the mean square error as evaluation metric for measuring the quantity of noise before and after denoising. Subfigure 5(a) show us the optimal threshold for trigonometric signals start are in range of $[0.8, 1]$, and the origial error (without action of denoiser) is equal to variance, 1. On the other hand, the polynomial signal type hit the best performance with threshold 0.05, and increasing threshold value leading hurting performance of model.

The last subfigure 5(c), on the IBM stock market dataset, visualized the signal to noise ratio metric over different threshold values. The optimal point is $2.1e^{-5}$ corresponding optimal value -32.8542, which improve 0.06% compared to the original SNR, -32.835.

D. Conclusion and future works

This project effectively applies the Fourier Transform (FT), specifically the Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT), in signal processing and data compression. It addresses the challenges of analyzing noisy data and implements an FT-based amplitude thresholding noise removal framework. The framework, tested on various 1D datasets, proves effective in reducing noise. However, the selection of the threshold value is critical for optimal performance.

For future work, there are several potential directions for extension. One could explore different methods for threshold selection, possibly incorporating machine learning techniques to dynamically determine the optimal threshold [18]. Additionally, the framework could be extended to handle multi-dimensional data, broadening its applicability [19].

REFERENCES

[1] Ronald N Bracewell. The fourier transform. *Scientific American*, 260(6):86–95, 1989.

[2] Duraisamy Sundararajan. *The discrete Fourier transform: theory, algorithms and applications*. World Scientific, 2001.

[3] Chakka Sri Harsha Kaushik, Tadvarthi Gautam, and V Elamaran. A tutorial review on discrete fourier transform with data compression application. In *2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, pages 1–6. IEEE, 2014.

[4] Nakhlé H Asmar. *Partial differential equations with Fourier series and boundary value problems*. Courier Dover Publications, 2016.

[5] M Farooq Wahab, Fabrice Gritti, and Thomas C O'Haver. Discrete fourier transform techniques for noise reduction and digital enhancement of analytical signals. *TrAC Trends in Analytical Chemistry*, 143:116354, 2021.

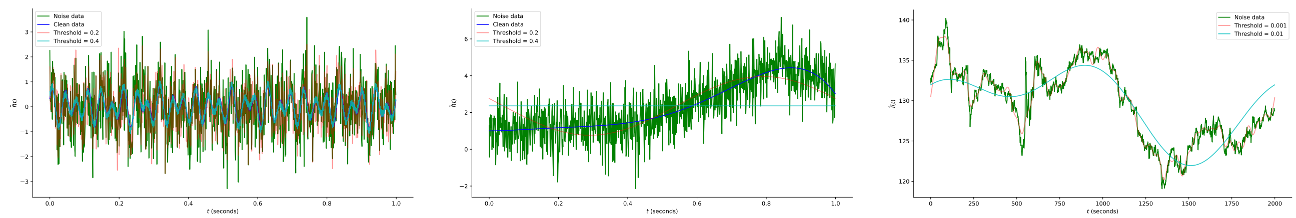
[6] Pierre Duhamel and Martin Vetterli. Fast fourier transforms: a tutorial review and a state of the art. *Signal processing*, 19(4):259–299, 1990.

[7] Hui Xiong, Gaurav Pandey, Michael Steinbach, and Vipin Kumar. Enhancing data analysis with noise removal. *IEEE transactions on knowledge and data engineering*, 18(3):304–319, 2006.

[8] Anders Brandt. *Noise and vibration analysis: signal analysis and experimental procedures*. John Wiley & Sons, 2023.

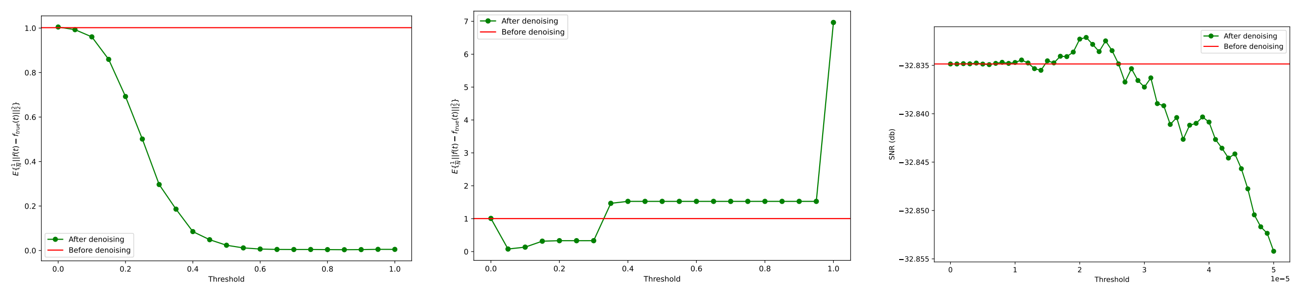
[9] Sang Gyu Kwak and Jong Hae Kim. Central limit theorem: the cornerstone of modern statistics. *Korean journal of anesthesiology*, 70(2):144, 2017.

[10] Zhang Yonglin and Zhang Jiafan. Numerical simulation of stochastic road process using white noise filtration. *Mechanical Systems and Signal Processing*, 20(2):363–372, 2006.



(a) (b) (c)

Fig. 4. Signal reconstruction results using FT-based denoising framework. The subfigure 4(a) is visualization of cleaned trigonometric signal with two different threshold 0.2, and 0.4; the subfigure 4(b) is for polynomial signal and the last subfigure 4(c) shows the reconstructed signal on IBM dataset input with two different threshold ratio $1e^{-3}$ and $1e^{-2}$. In 3 cases, the blue curves (if available) denote for the true signal w/o noise, the green curves stand for acquired signals which need to be denoised, the red curve denotes for threshold 0.2 (or $1e^{-3}$ for subfigure 4(c)), while the result of threshold 0.4 (or $1e^{-2}$ for subfigure 4(c)) is denoted as cyan curves.



(a) (b) (c)

Fig. 5. Optimal values for three types of dataset. The subfigure 5(a), 5(b), 5(c) show the results on trigonometric, polynomial and IBM dataset respectively. The horizontal red lines denote for the metric of original input signal (with noise), while the green curves show performance of FT-based denoising framework varied on different threshold ratios.

[11] MA Josephine Sathya and SP Victor. Noise reduction techniques and algorithms for speech signal processing. *International Journal of Scientific Engineering and Research*, 6(1), 2015.

[12] Don DeLillo. *White noise*. Penguin, 1999.

[13] Hui-Hsiung Kuo. *White noise distribution theory*. CRC press, 2018.

[14] David Brandwood. *Fourier transforms in radar and signal processing*. Artech House, 2012.

[15] Seema Sud. Noise and interference suppression in sonar echoes using the fractional fourier transform. *European Journal of Engineering and Technology Research*, 2(7):40–44, 2017.

[16] Ann Maria John, Kiran Khanna, Ritika R Prasad, and Lakshmi G Pillai. A review on application of fourier transform in image restoration. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 389–397. IEEE, 2020.

[17] Don H Johnson. Signal-to-noise ratio. *Scholarpedia*, 1(12):2088, 2006.

[18] Jesús A De Loera, Jamie Haddock, Anna Ma, and Deanna Needell. Data-driven algorithm selection and tuning in optimization and signal processing. *Annals of Mathematics and Artificial Intelligence*, 89(7):711–735, 2021.

[19] Aryan Govil, Eric Yao, and Christina R Borao. Multi-dimensional framework for eeg signal processing and denoising through tensor-based architecture. *arXiv preprint arXiv:2401.05589*, 2024.

Appendix A
Proof

A. Proof of the mean square error of synthetic signal is equal to variance of noise

Appendix B
Repository

A. IBM dataset

B. Code