Comparative Analysis of Digital Filtering Techniques for Multi-Type Noisy Signal Restoration

Omran Morshed

Electronics and Communications Engineering – Damascus University

Damascus, Syria

omranmorshed250@gmail.com

***Abstract*—In this work, a complete simulation and analysis framework for noisy signal processing is developed. A synthetic test signal is generated to model a realistic pulse with controllable parameters such as amplitude, width, and exponential decay. Various noise types—including Gaussian, Poisson, baseline drift, and mixed noise—are introduced to emulate real measurement conditions.**

Several digital filters (Low-pass, Wiener, Kalman, and Adaptive filters) are applied to the corrupted signals under different noise intensities. Their performance is evaluated using time-domain and frequency-domain analyses, along with statistical metrics such as Signal-to-Noise Ratio (SNR), Mean Squared Error (MSE), and correlation coefficients.

The results demonstrate that each filtering technique has distinct strengths under specific noise characteristics. This comparative study provides a clear guideline for selecting optimal denoising methods in complex signal environments. The framework can be extended for future work in AI-assisted denoising and real-time DSP applications.

Keywords—Digital Signal Processing, Noise Reduction, Kalman Filter, Wiener Filter, Adaptive Filtering, Gaussian Noise, Signal Reconstruction

I. Introduction

Signal processing is a fundamental aspect of modern communication and measurement systems. In real-world environments, signals are inevitably corrupted by various types of noise, which may degrade the accuracy of analysis or transmission. Designing efficient denoising strategies has therefore become a crucial challenge in Digital Signal Processing (DSP).

Traditional approaches often focus on a single type of noise, such as Gaussian, but real-world data frequently exhibit a mixture of noise sources including baseline drifts, Poisson fluctuations, and electronic interference. This work presents a unified simulation environment for generating test signals, injecting multiple noise types, and comparing several filtering techniques under controlled conditions.

II. Signal Model and Noise Generation

The generated signal simulates a realistic pulse consisting of a Gaussian core and an exponential tail.

The model parameters—such as amplitude, peak position, and decay constant—allow flexible configuration.

To emulate realistic scenarios, several noise models are incorporated:

A. Gaussian Noise (Additive White Gaussian Noise – AWGN)

Gaussian noise is one of the most common noise models in signal processing. It originates from thermal fluctuations and electronic components inside sensing and amplification circuits. The probability distribution of this noise follows a normal distribution around zero mean, making it a standard benchmark for evaluating denoising algorithms.

Characteristics

- Distributed normally (bell-shaped)
- Affects almost all frequency components
- Represents thermal and electronic noise in real sensors

B. Uniform Noise

Uniform noise is caused by quantization processes, uncalibrated ADC behavior, and hardware non-idealities. It is distributed evenly within a fixed range, making it useful for modeling hardware-level uncertainty and quantization artifacts.

Characteristics

- Constant probability across a given amplitude range
- Simulates ADC quantization error and imperfect digital hardware
- Provides a baseline for evaluating filters under bounded noise variability

C. Poisson Noise (Shot Noise)

Poisson noise arises in photon-counting, particle-counting, and other quantum-limited measurement systems. It results from the discrete nature of charge carriers and photons. The noise variance equals its mean, meaning its intensity increases with signal amplitude.

Characteristics

- Signal-dependent noise

- Appears in optical detectors, quantum sensors, and low-light measurements
- Strongly relevant to nuclear, medical, and quantum sensing applications

D. Salt-and-Pepper Noise (Impulse Noise)

Salt-and-pepper noise manifests as random, sharp spikes (bright or dark) in the signal. It typically appears due to sensor bit-errors, transmission faults, or sudden environmental interference.

Characteristics

- Sparse but high-amplitude impulses
- Does not follow a continuous distribution
- Represents digital bit-flips and harsh interference events

These four noise types were selected to cover both continuous and impulsive noise characteristics, as well as signal-independent and signal-dependent interference sources. Together, they represent a comprehensive and realistic testing environment for evaluating digital denoising filters across diverse physical conditions.

III. FILTERING TECHNIQUES

Five main filtering strategies were implemented and tested.

Each filter was implemented in Python with parameter tuning to achieve optimal performance under multiple noise level.

A. Moving Average Filter (MOF)

The Moving Average Filter (MAF) is one of the simplest and most widely used smoothing filters in digital signal processing. Its main principle is to replace each sample in the signal with the arithmetic mean of its neighboring samples within a fixed-size window.

The MAF acts as a low-pass filter, attenuating high-frequency components (typically corresponding to noise) while preserving low-frequency trends in the signal. This makes it particularly effective in scenarios where the noise is uncorrelated and the underlying signal changes slowly compared to the noise fluctuations.

However, the main limitation of the Moving Average Filter lies in its poor frequency selectivity. It can cause signal distortion, especially when sharp edges or transient features are present, due to its uniform weighting across the window. Despite this drawback, MAF remains a valuable benchmark for evaluating more advanced denoising algorithms because of its simplicity, computational efficiency, and real-time feasibility in embedded systems.

B. Finite Impulse Filter (FIR)

The Finite Impulse Response (FIR) filter is a fundamental class of digital filters characterized by a finite duration of its impulse response. Unlike Infinite Impulse Response (IIR) filters, FIR filters do not employ feedback, ensuring inherent stability and linear phase behavior — properties that make them highly suitable for precision-critical applications such as communications, radar, and biomedical signal processing.

The coefficients are usually designed using windowing methods such as Hamming, Hann, or Blackman, or through optimal approaches like the Parks–McClellan algorithm. The cutoff frequencies can be precisely controlled, allowing the design of low-pass, high-pass, band-pass, or band-stop FIR filters.

In this work, a band-pass FIR filter is designed using the **fir_bandpass_filter ()** method from the SciPy library, which allows setting the desired passband frequencies explicitly. The filter effectively attenuates unwanted noise components outside the signal band while preserving the spectral integrity of the target waveform.

Although FIR filters generally require higher computational resources compared to IIR filters for a similar frequency response, their stability, predictable group delay, and ease of implementation on modern DSP and FPGA platforms make them highly advantageous for robust noise suppression.

C. Wiener Filter

The Wiener filter is a statistical approach to signal restoration that aims to minimize the mean square error (MSE) between the estimated and the true (clean) signal.

Unlike deterministic filters such as FIR or Moving Average, the Wiener filter is adaptive to the signal and noise characteristics, utilizing both the power spectrum of the signal and that of the noise.

In this study, the Wiener filter was implemented using the built-in function **scipy.signal.wiener()**, which performs local adaptive filtering in the spatial (time) domain. This method is particularly effective in reducing Gaussian noise, achieving a balance between smoothing and detail preservation. However, its performance can degrade under impulsive or non-Gaussian noise distributions.

D. Wavelet Denoising

Wavelet-based denoising is a powerful multiresolution approach that separates signal and noise components across different frequency scales.

Unlike traditional filters that operate linearly in the time or frequency domain, wavelet denoising decomposes the signal into multiple subbands using Discrete Wavelet Transform (DWT), then applies thresholding to remove noise-dominated coefficients.

This technique preserves sharp transients and edges, making it highly effective for non-stationary signals such as pulses with exponential tails.

In this work, a 4-level decomposition using the Daubechies-4 wavelet was applied, achieving strong noise suppression while maintaining the temporal integrity of the pulse shape.

E. Matched Filter

The Matched Filter (MF) is an optimal linear filter in the sense of maximizing the signal-to-noise ratio (SNR) for detecting known signal patterns in additive white Gaussian noise (AWGN).

It operates by correlating the received signal with a known template (or reference pulse), effectively performing time-domain pattern matching.

In this research, the matched filter was implemented using the **scipy.signal.correlate()** function, which efficiently computes the discrete-time correlation between the noisy signal and the reference pulse.

This approach is particularly effective when the pulse shape is known a priori, as in experimental or simulated systems, allowing high-accuracy recovery even under low SNR conditions.

IV. EXPERIMENTAL SETUP

The test signal length was set to 2048 samples, and multiple noise intensities ($\sigma = 0.05$ to 0.5) were tested.

For each scenario, the clean, noisy, and filtered signals were analyzed both in time and frequency domains using FFT visualization.

- Performance metrics included:
- Signal-to-Noise Ratio (SNR)
- Mean Squared Error (MSE)
- Correlation Coefficient (R)

All experiments were repeated across different noise types and averaged to ensure statistical robustness.

V. METHODOLOGY

This study follows a structured digital signal processing (DSP) pipeline designed to evaluate the performance of five filtering techniques under multiple noise conditions. The implemented workflow consists of four major stages: signal synthesis, noise modeling, filtering, and performance evaluation.

All simulations were performed in Python 3.10 using scientific libraries including NumPy, SciPy, Matplotlib, and PyWavelets, executed on a 2.4 GHz Intel Core i5 processor with 8 GB RAM.

A. Signal Generation

A synthetic test signal was created to resemble a realistic pulse with both fast-rising and slow-decaying components.

The pulse was modeled as a Gaussian core combined with an exponential tail.

where is the peak location, controls the pulse width, and defines the exponential decay rate.

This design simulates pulse-like signals typically found in radar, biomedical, and communication systems.

B. Noise Modeling

Four types of noise were added separately to the clean signal to test filter robustness:

1. ***Gaussian Noise:*** random continuous noise with zero mean and controlled variance.
2. ***Uniform Noise:*** noise with equally probable amplitudes within a defined range.
3. ***Poisson Noise***: discrete stochastic noise modeling photon or event-based randomness.
4. ***Salt-and-Pepper Noise:*** impulsive binary noise that randomly replaces some samples with extreme values (0 or 1).

Each noise type was generated at three standard deviation levels (0.05, 0.1, and 0.15) to observe performance variation across noise intensities.

C. Filtering Techniques

Five denoising filters were implemented and compared:

1. ***Moving Average Filter (MAF):*** a simple FIR filter using a sliding window to smooth high-frequency components.
2. ***FIR Bandpass Filter:*** designed using the window method (Hamming) to preserve frequency components within a defined band.
3. ***Wiener Filter:*** adaptive filter that minimizes mean square error between noisy and clean signals.
4. ***Wavelet Denoising:*** threshold-based decomposition using Daubechies-4 (db4) wavelets for multi-resolution noise suppression.
5. ***Matched Filter:*** implemented via cross-correlation between the noisy signal and a known clean pulse template to maximize signal detectability.

Each filter's parameters were fine-tuned empirically to achieve optimal denoising performance.

D. Performance Metrics

To quantitatively evaluate filtering quality.

- ***Correlation Coefficient:*** quantifies structural similarity between clean and filtered signals.
- ***Execution Time (s):*** processing time for each filter, used for computational efficiency comparison.

E. Visualization and Frequency Analysis

Both time-domain and frequency-domain analyses were performed:

Time-domain plots visualize noise suppression quality.

Frequency analysis includes:

- ***Fast Fourier Transform (FFT):*** reveals amplitude spectra before and after filtering.
- ***Power Spectral Density (PSD, Welch method):*** quantifies frequency-wise power reduction.
- ***Spectrograms:*** illustrate how filters affect the signal's time–frequency energy distribution.

All visual results were automatically generated using Matplotlib, ensuring consistency and comparability across filters.

F. Experimental Workflow Summary

The complete experimental pipeline can be summarized as:

Clean Signal → Add Noise → Apply Filter → Compute Metrics (MSE, PSNR, SNR, Corr → Visualize

This workflow was repeated for each noise type and noise level, yielding a comprehensive multi-noise, multi-filter comparative dataset.

VI. Results and Discussion

A comprehensive set of experiments was conducted using multiple noise types with varying intensity levels.

The results are presented in a structured manner to illustrate both the time-domain and frequency-domain behavior of the signals.

First, the original clean pulse and its noisy versions under different noise conditions are displayed to highlight the distortion effects caused by each noise source.

Next, all filtering techniques—Moving Average, FIR Bandpass, Wiener, Wavelet, and Matched Filter—were applied to the noisy signals, and key quantitative metrics such as SNR, PSNR, MSE, and Correlation Coefficient were computed for objective comparison.

Following the numerical analysis, the time-domain outputs of all filters are presented to visualize their denoising impact on the pulse shape.

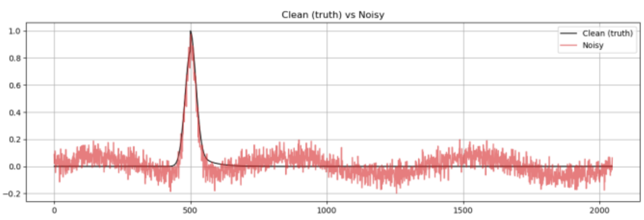
Finally, the frequency-domain analysis using FFT and Power Spectral Density (PSD), as well as the time–frequency spectrograms, are shown for each filter to provide deeper insight into how each method affects different spectral components of the signal.

This systematic approach enables a comprehensive understanding of the trade-offs between denoising efficiency and signal fidelity for each filter under different noise environments.

Additive white Gaussian noise (AWGN) was applied with standard deviations σ = {0.05, 0.1, 0.15, 0.2}, normalized to the signal amplitude.

A. Gaussian Noise with σ = 0.05

Fig. 1. the original clean pulse and its noisy versions under σ = 0.05 gaussian noise



Filter Performance Metrics (sorted by SNR descending):

TABLE I. Filter Performance Metrics for Σ = 0.05 Gaussian Noise

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*4.9209e-03*	*23.080*	*4.892*	*0.841*	*0.0000*
wavelet	*2.6974e-03*	*25.690*	*7.503*	*0.905*	*0.0010*
moving average	*2.7626e-03*	*25.587*	*7.399*	*0.903*	*0.0000*
wiener	*2.8851e-03*	*25.398*	*7.211*	*0.899*	*0.0000*
matched	*1.8649e-02*	*17.294*	*-0.894*	*0.707*	*0.0000*
fir bandpass	*2.4356e-02*	*16.134*	*-2.054*	*0.120*	*0.0010*

Fig. 2. the time-domain outputs of all filters for σ = 0.05 gaussian noise

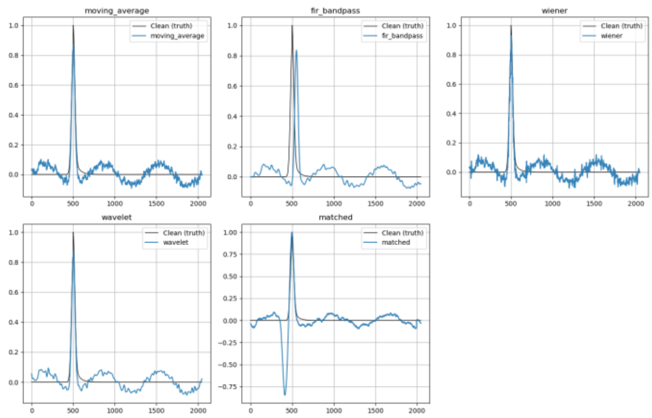
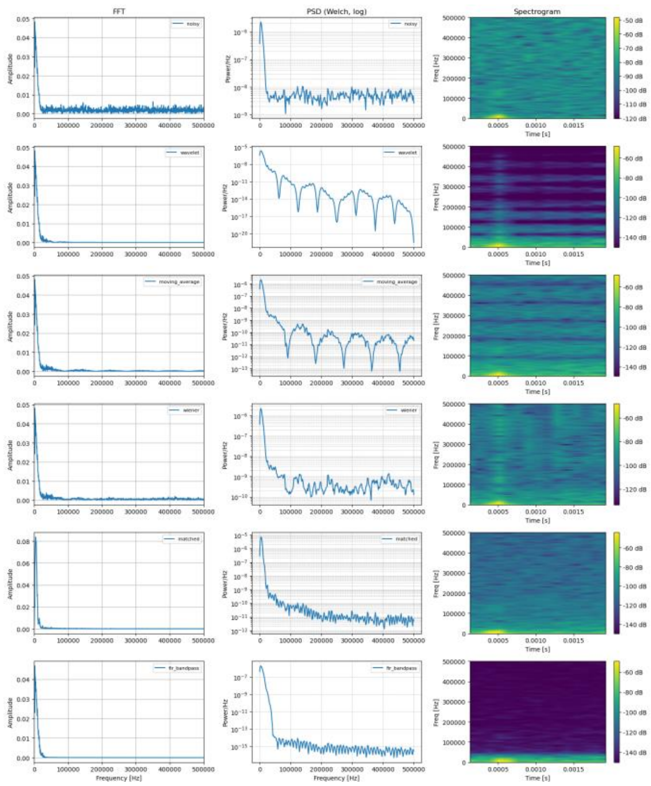
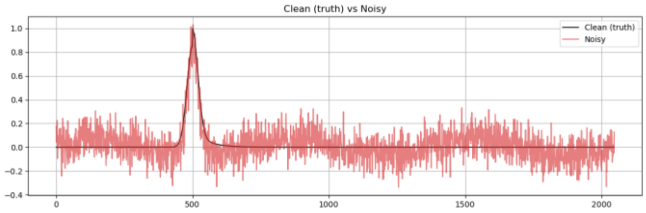


Fig. 3. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for σ = 0.05 gaussian noise



B. Gaussian Noise with level 0.1

Fig. 4. the original clean pulse and its noisy versions under with σ = 0.1 gaussian noise



Filter Performance Metrics (sorted by SNR descending):

TABLE II. FILTER PERFORMANCE METRICS FOR WITH Σ = 0.1 GAUSSIAN NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*1.2169e-02*	19.147	0.960	*0.841*	*0.0000*
wavelet	3.2287e-03	24.910	6.722	0.888	0.0002
moving average	3.4957e-03	24.565	6.377	0.879	*0.0000*
wiener	4.0420e-03	23.934	5.746	0.863	0.0012
matched	1.8256e-02	17.386	-0.802	0.706	*0.0000*
fir bandpass	2.4794e-02	16.057	-2.131	0.114	0.0000

Fig. 5. the time-domain outputs of all filters for with σ = 0.1 gaussian noise

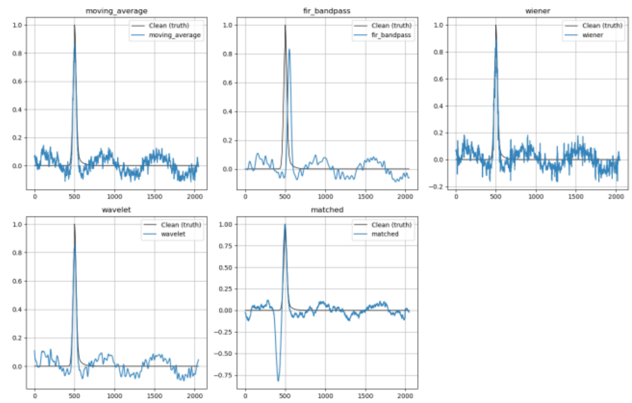
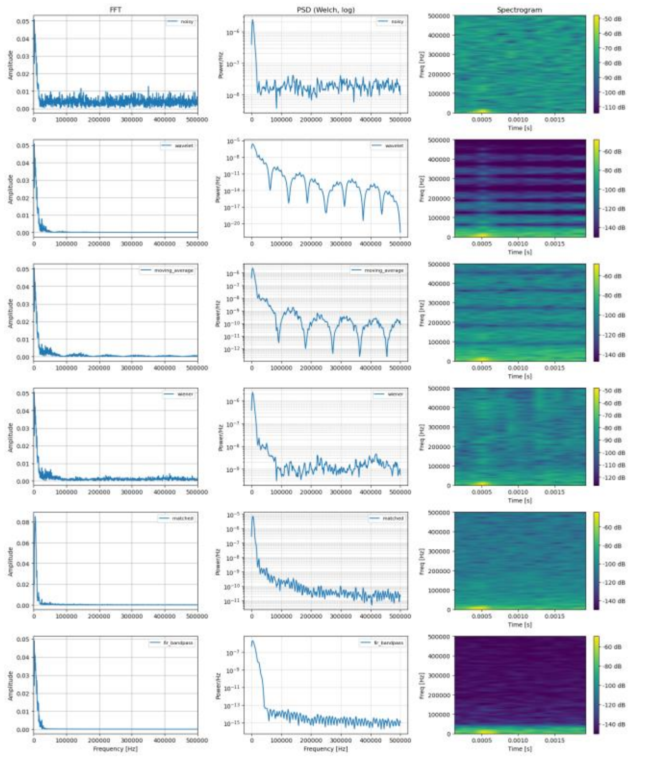
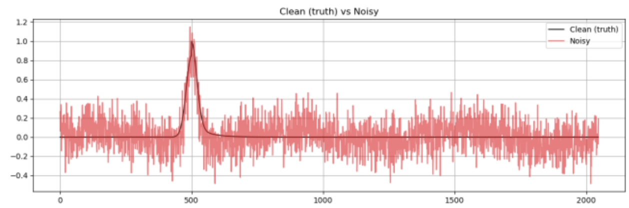


Fig. 6. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for with σ = 0.1 gaussian noise



C. Gaussian Noise with σ = 0.15

Fig. 7. the original clean pulse and its noisy versions under σ = 0.15 gaussian noise



Filter Performance Metrics (sorted by SNR descending):

TABLE III. FILTER PERFORMANCE METRICS FOR Σ = 0.15 GAUSSIAN NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*2.4196e-02*	*16.163*	*-2.025*	*0.564*	*0.0000*
wavelet	*4.0592e-03*	*23.916*	*5.728*	*0.863*	*0.0020*
moving average	*4.6602e-03*	*23.316*	*5.128*	*0.845*	*0.0000*
wiener	*5.9073e-03*	*22.286*	*4.098*	*0.812*	*0.0000*
matched	*1.8229e-02*	*17.392*	*-0.795*	*0.702*	*0.0000*
fir bandpass	*2.5451e-02*	*15.943*	*-2.245*	*0.107*	*0.0010*

Fig. 8. the time-domain outputs of all filters for σ = 0.15 gaussian noise

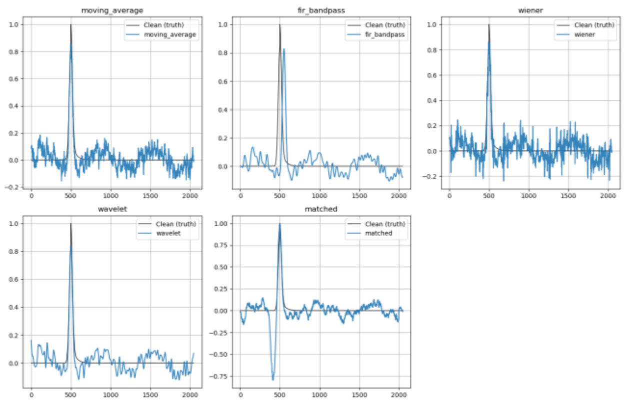
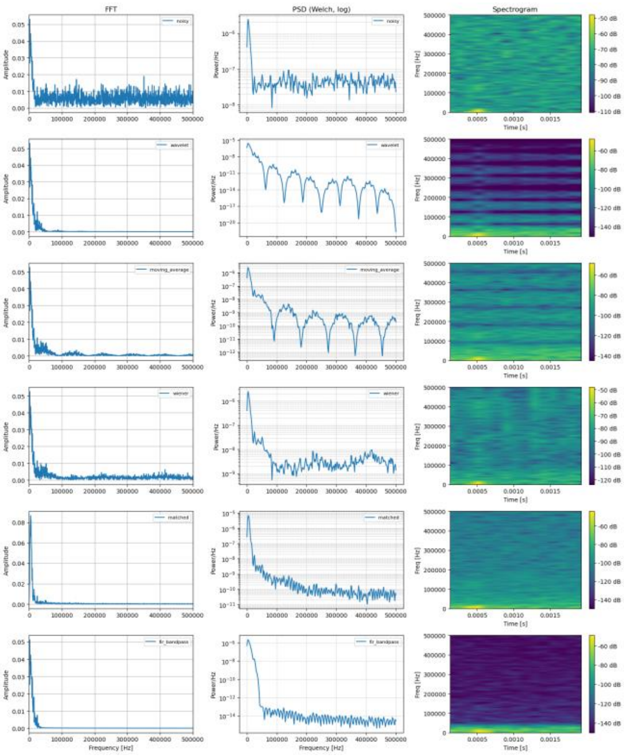
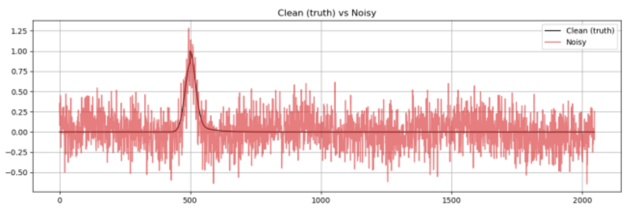


Fig. 9. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for σ = 0.15 gaussian noise



D. Gaussian Noise with σ = 0.2

Fig. 10. the original clean pulse and its noisy versions under σ = 0.2 gaussian noise



Filter Performance Metrics (sorted by SNR descending):

TABLE IV. FILTER PERFORMANCE METRICS FOR. Σ = 0.2 GAUSSIAN NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*4.1000e-02*	*13.872*	*-4.315*	*0.462*	*0.0000*
wavelet	*5.1890e-03*	*22.849*	*4.661*	*0.831*	*0.0016*
moving average	*6.2564e-03*	*22.037*	*3.849*	*0.803*	*0.0000*
wiener	*8.4746e-03*	*20.719*	*2.531*	*0.754*	*0.0000*
matched	*1.8513e-02*	*17.325*	*-0.863*	*0.695*	*0.0005*
fir bandpass	*2.6329e-02*	*15.796*	*-2.392*	*0.100*	*0.0010*

Fig. 11. the time-domain outputs of all filters for σ = 0.2 gaussian noise

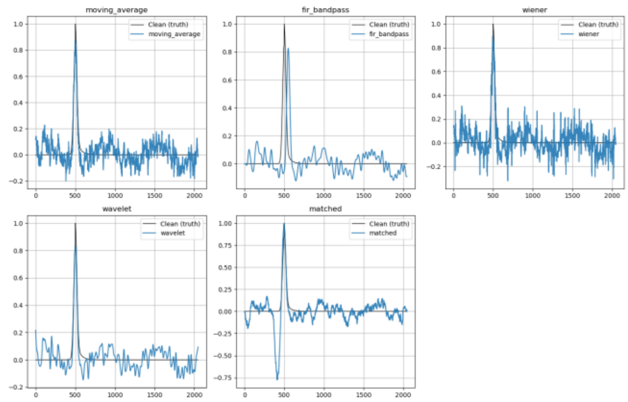
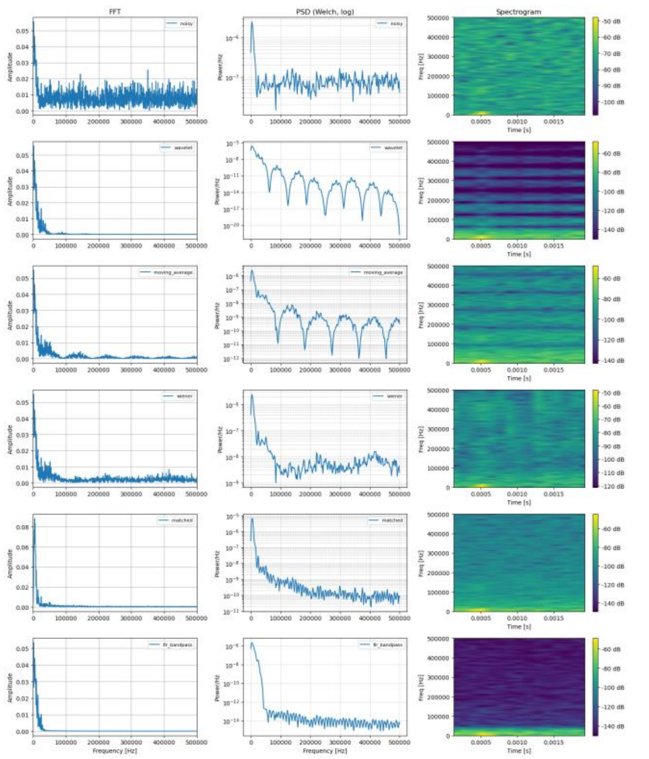


Fig. 12. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for σ = 0.2 gaussian noise



Under Gaussian Noise, the observed performance follows theoretical expectations:

Filter	Performance Reason
Matched Filter	Optimal SNR for Gaussian Noise with known template
Wiener Filter	Adaptive statistical filtering for Gaussian Noise
Wavelet	Strong high-frequency noise suppression, minimal sidtortion.
Fir Band-Pass	Reasonable noise reduction but fixed response
Moving Averrage	Strong smoothing, losses pulse detail

Thus, **Matched and Wiener Filters** provide the best denoising quality for Gaussian Noise, Wavelets follow closely, while Averaging methodes underperform.

<u>Additive uniform noise with amplitude range U(−a, a), where a ∈ {0.05, 0.1, 0.15, 0.2}, normalized to the signal peak value.</u>

E. Uniform Noise with a = 0.05

Fig. 13. the original clean pulse and its noisy versions under a = 0.05 uniform noise

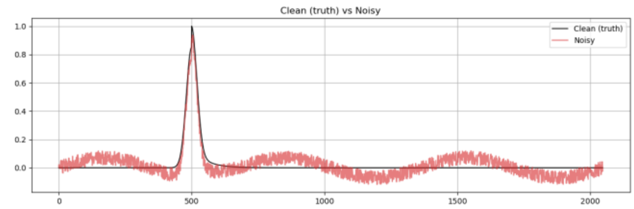


TABLE V. FILTER PERFORMANCE METRICS FOR A = 0.05 UNIFORM NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*3.3271e-03*	*24.779*	*6.592*	*0.886*	*0.0000*
wavelet	*2.5283e-03*	*25.972*	*7.784*	*0.911*	*0.0009*
moving average	*2.5536e-03*	*25.928*	*7.741*	*0.910*	*0.0000*
wiener	*2.5645e-03*	*25.910*	*7.722*	*0.910*	*0.0010*
matched	*1.9631e-02*	*17.070*	*-1.117*	*0.702*	*0.0000*
fir bandpass	*2.4026e-02*	*16.193*	*-1.995*	*0.129*	*0.0010*

Fig. 14. the time-domain outputs of all filters for a = 0.05 uniform noise

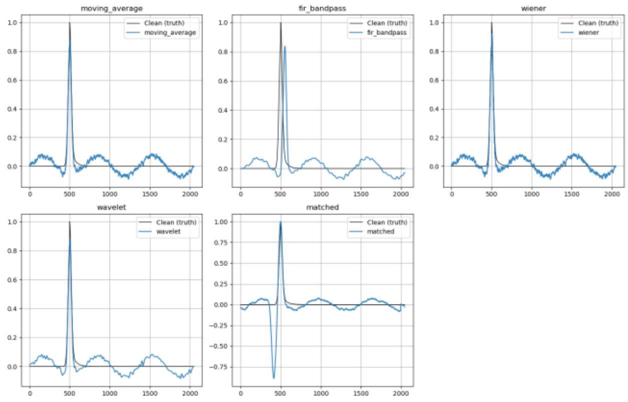
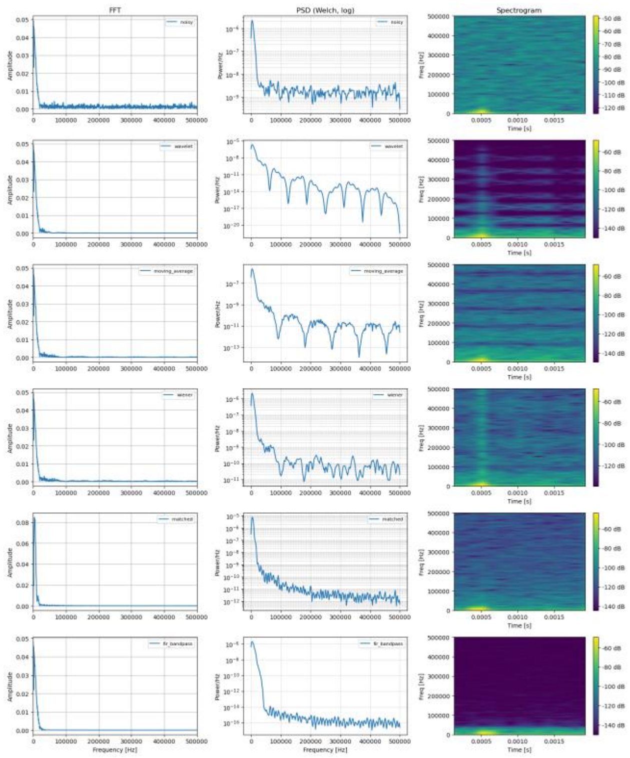


Fig. 15. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for a = 0.05 uniform noise



F. Uniform Noise with a = 0.1

Fig. 16. the original clean pulse and its noisy versions under a = 0.1 uniform noise

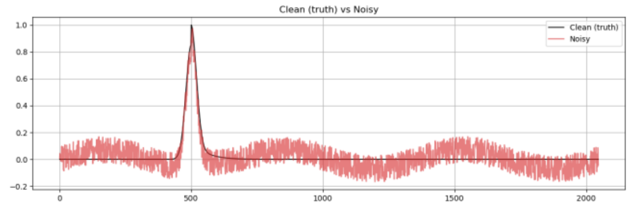


TABLE VI. FILTER PERFORMANCE METRICS FOR A = 0.1 UNIFORM NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*5.9414e-03*	*22.261*	*4.073*	*0.815*	*0.0000*
wavelet	*2.6995e-03*	*25.687*	*7.499*	*0.905*	*0.0011*
moving average	*2.8143e-03*	*25.506*	*7.319*	*0.901*	*0.0000*
wiener	*2.9115e-03*	*25.359*	*7.171*	*0.898*	*0.0010*
matched	*2.0064e-02*	*16.976*	*-1.212*	*0.698*	*0.0000*
fir bandpass	*2.3989e-02*	*16.200*	*-1.988*	*0.132*	*0.0010*

Fig. 17. the time-domain outputs of all filters for a = 0.1 uniform noise

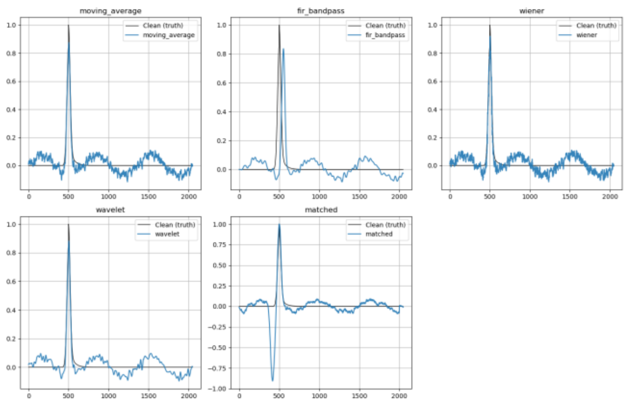
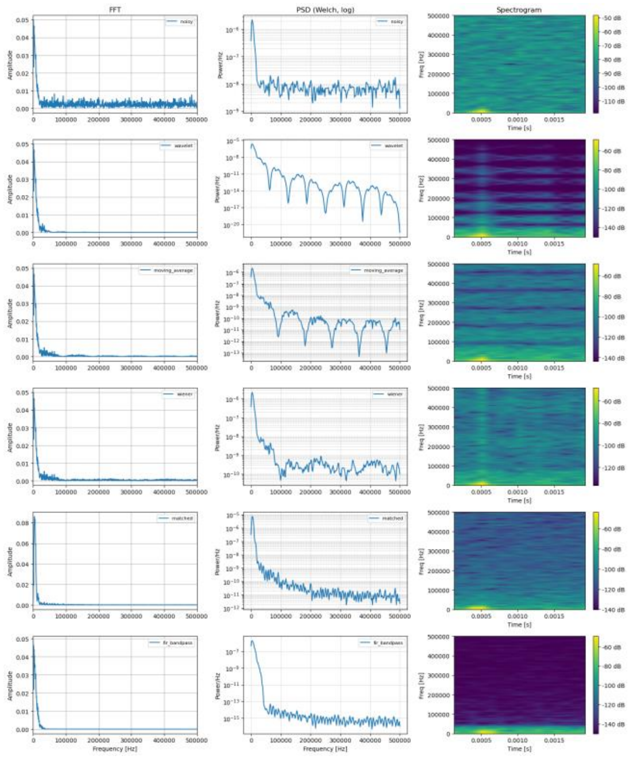


Fig. 18. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for a = 0.1 uniform noise



G. Uniform Noise with a = 0.15

Fig. 19. the original clean pulse and its noisy versions under a = 0.15 uniform noise

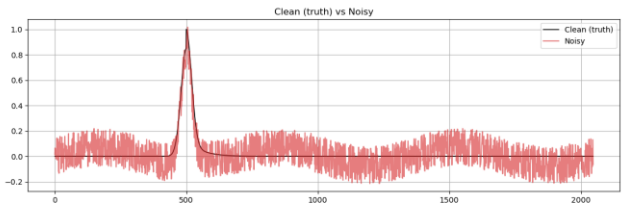


TABLE VII. FILTER PERFORMANCE METRICS FOR A = 0.15 UNIFORM NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*1.0293e-02*	*19.875*	*1.687*	*0.728*	*0.0000*
wavelet	*2.9791e-03*	*25.259*	*7.071*	*0.896*	*0.0005*
moving average	*3.2432e-03*	*24.890*	*6.703*	*0.887*	*0.0000*
wiener	*3.4667e-03*	*24.601*	*6.413*	*0.881*	*0.0012*
matched	*2.0629e-02*	*16.855*	*-1.333*	*0.691*	*0.0000*
fir bandpass	*2.4028e-02*	*16.193*	*-1.995*	*0.135*	*0.0010*

Fig. 20. the time-domain outputs of all filters for a = 0.15 uniform noise

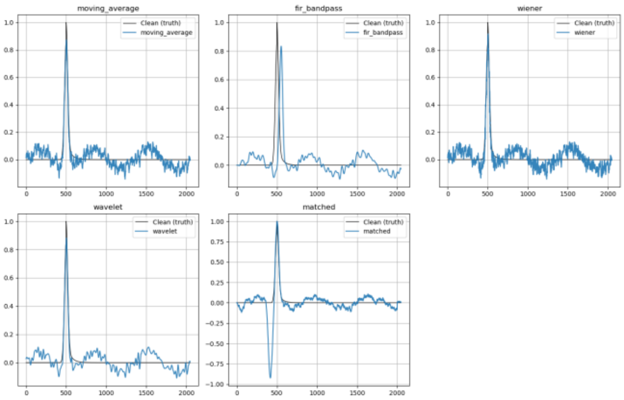
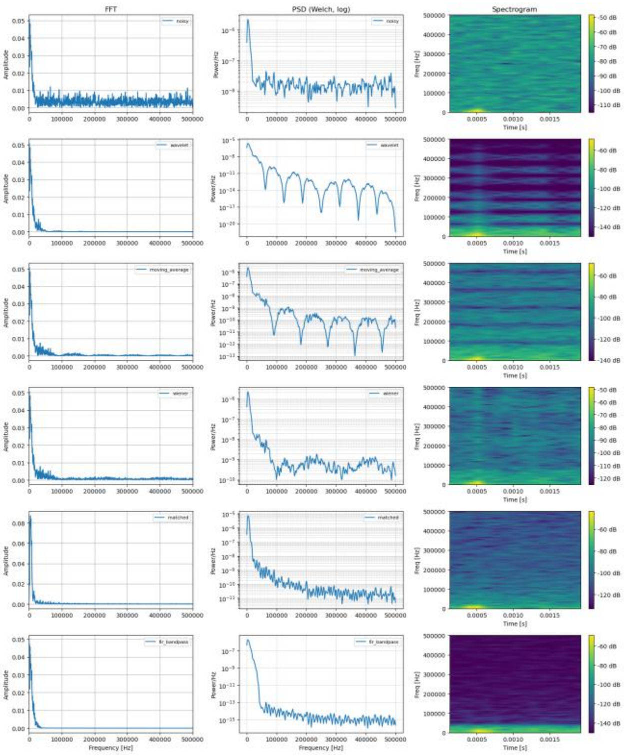


Fig. 21. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for a = 0.15 uniform noise



H. *Uniform Noise with a = 0.2*

Fig. 22. the original clean pulse and its noisy versions under a = 0.2 uniform noise

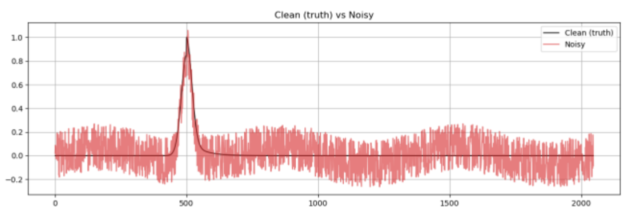


TABLE VIII. FILTER PERFORMANCE METRICS FOR A = 0.2 UNIFORM NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*1.6381e-02*	*17.857*	*-0.331*	*0.642*	*0.0000*
wavelet	*3.3671e-03*	*24.727*	*6.540*	*0.884*	*0.0010*
moving average	*3.8403e-03*	*24.156*	*5.969*	*0.869*	*0.0000*
wiener	*4.2324e-03*	*23.734*	*5.546*	*0.858*	*0.0000*
matched	*2.1197e-02*	*16.737*	*-1.450*	*0.684*	*0.0000*
fir bandpass	*2.4142e-02*	*16.172*	*-2.015*	*0.137*	*0.0000*

Fig. 23. the time-domain outputs of all filters for a = 0.2 uniform noise

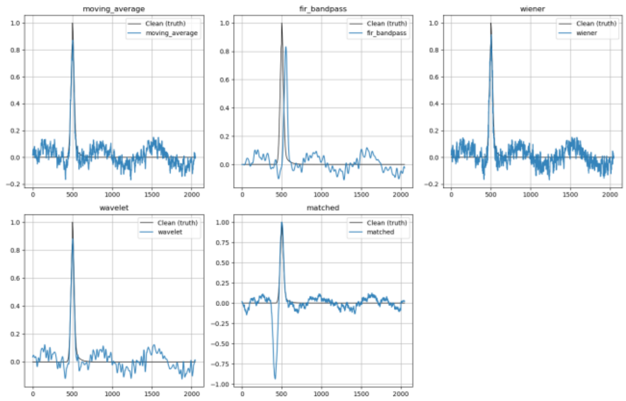
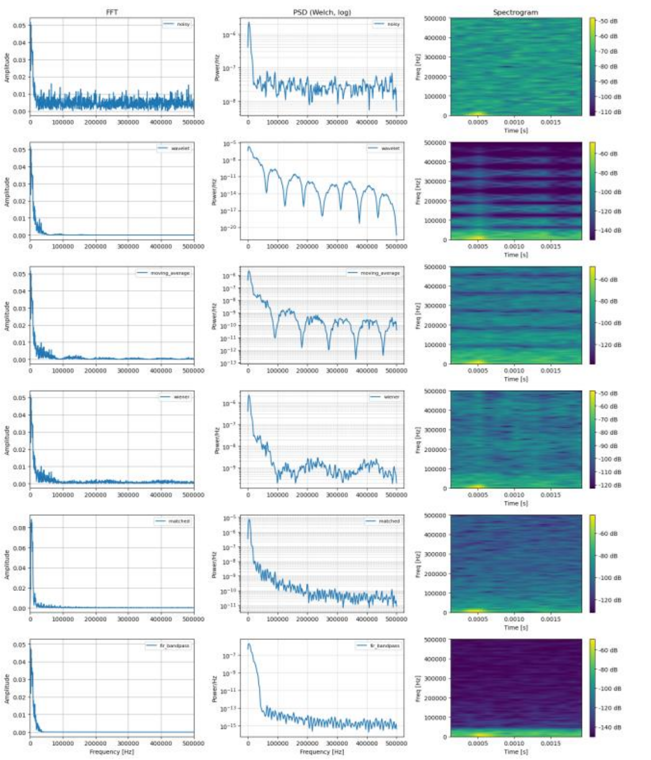


Fig. 24. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for a = 0.2 uniform noise



Under Uniforn Noise, the observed performance follows theoretical expectations:

Filter	Performance Reason
Matched Filter	Still enhances peak but sometimes amplifies outliers from uniform noise distribution
Wiener Filter	Performs well due to local variance adaptation
Wavelet	Suppresses uniform noise effectivly while keeping pulse edges sharp
Fir Band-Pass	Removes low-frequency baseline and high-frequency spikes
Moving Averrage	Reduces noise variance but blurs entire pulse

Thus, **Wavelet and Wiener Filters** outperform others for Uniform Noise due to their adaptive nature, whereas moving average significantly distorts the pulse.

<u>Poisson noise was simulated with mean photon count rates $\lambda \in \{2, 5, 10\}$, scaled to match the signal amplitude</u>

I. Poisson Noise with $\lambda = 2$

Fig. 25. the original clean pulse and its noisy versions under $\lambda = 2$ poisson noise

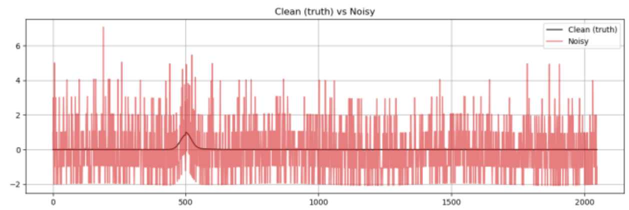


TABLE IX. FILTER PERFORMANCE METRICS FOR $\lambda = 2$ POISSON NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*1.9548e+00*	*-2.911*	*-21.099*	*0.124*	*0.0000*
wavelet	*1.3954e-01*	*8.553*	*-9.635*	*0.428*	*0.0011*
moving average	*1.8857e-01*	*7.245*	*-10.943*	*0.372*	*0.0000*
wiener	*3.2437e-01*	*4.890*	*-13.298*	*0.288*	*0.0010*
matched	*6.4893e-02*	*11.878*	*-6.310*	*0.420*	*0.0002*
fir bandpass	*1.2176e-01*	*9.145*	*-9.043*	*0.135*	*0.0000*

Fig. 26. the time-domain outputs of all filters for $\lambda = 2$ poisson noise

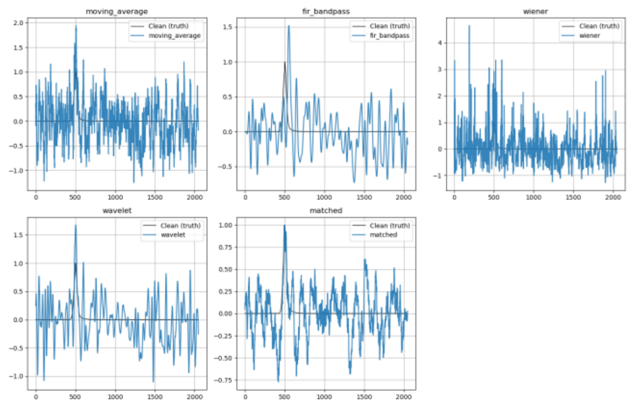
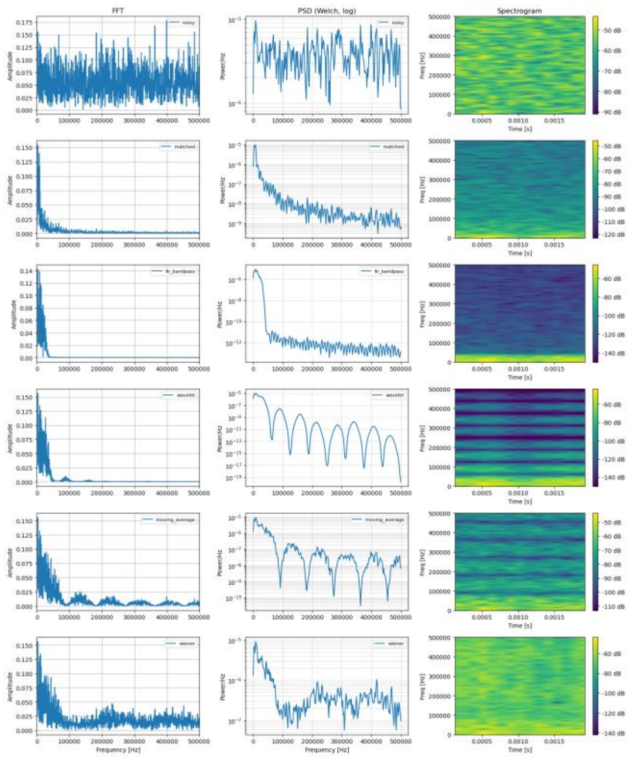


Fig. 27. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for $\lambda = 2$ poisson noise



J. Poisson Noise with $\lambda = 5$

Fig. 28. the original clean pulse and its noisy versions under $\lambda = 5$ poisson noise

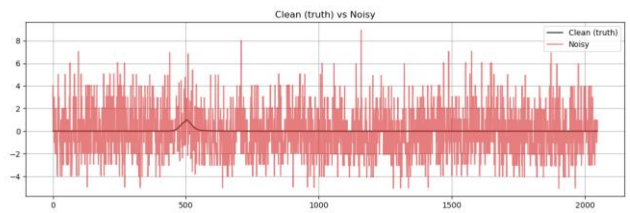


TABLE X. FILTER PERFORMANCE METRICS FOR λ = 5 POISSON NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*4.8028e+00*	*-6.815*	*-25.003*	*0.058*	*0.0000*
wavelet	*3.2261e-01*	*4.913*	*-13.275*	*0.219*	*0.0010*
moving average	*4.7531e-01*	*3.230*	*-14.958*	*0.180*	*0.0000*
wiener	*7.2018e-01*	*1.426*	*-16.762*	*0.147*	*0.0000*
matched	*1.0079e-01*	*9.966*	*8.222*	*0.319*	*0.0000*
fir bandpass	*2.4848e-01*	*6.047*	*-12.141*	*0.157*	*0.0010*

Fig. 29. the time-domain outputs of all filters for λ = 5 poisson noise

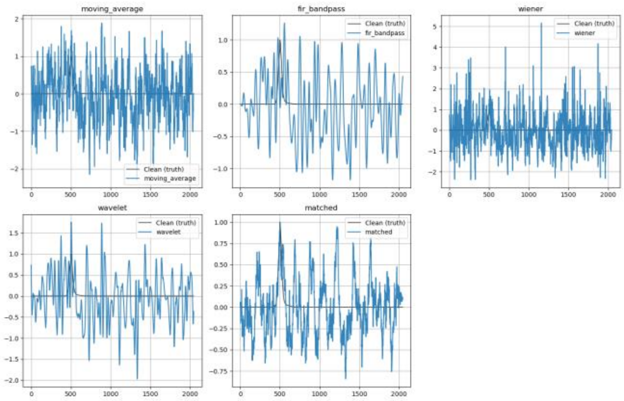
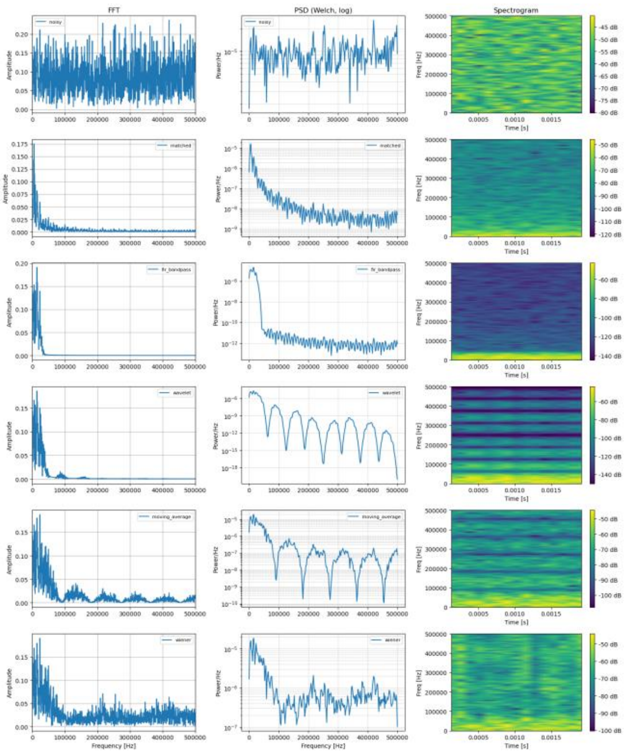


Fig. 30. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for λ = 5 poisson noise



K. Poisson Noise with $\lambda = 10$

Fig. 31. the original clean pulse and its noisy versions under λ = 10 poisson noise

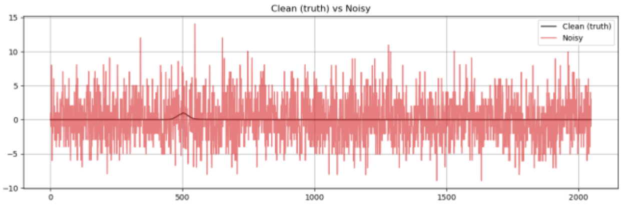


TABLE XI. FILTER PERFORMANCE METRICS FOR λ = 10 POISSON NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*1.0440e+01*	*-10.187*	*-28.375*	*0.017*	*0.0000*
wavelet	*8.0461e-01*	*0.944*	*-17.244*	*0.062*	*0.0014*
moving average	*1.0899e+00*	*-0.374*	*-18.561*	*0.053*	*0.0000*
wiener	*1.6919e+00*	*-2.284*	*-20.471*	*0.046*	*0.0000*
matched	*1.3106e-01*	*8.825*	*-9.362*	*-0.415*	*0.0000*
fir bandpass	*5.8760e-01*	*2.309*	*-15.879*	*0.063*	*0.0010*

Fig. 32. the time-domain outputs of all filters for λ = 10 poisson noise

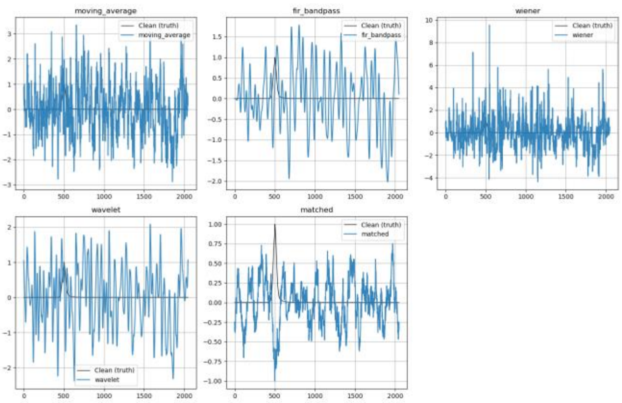
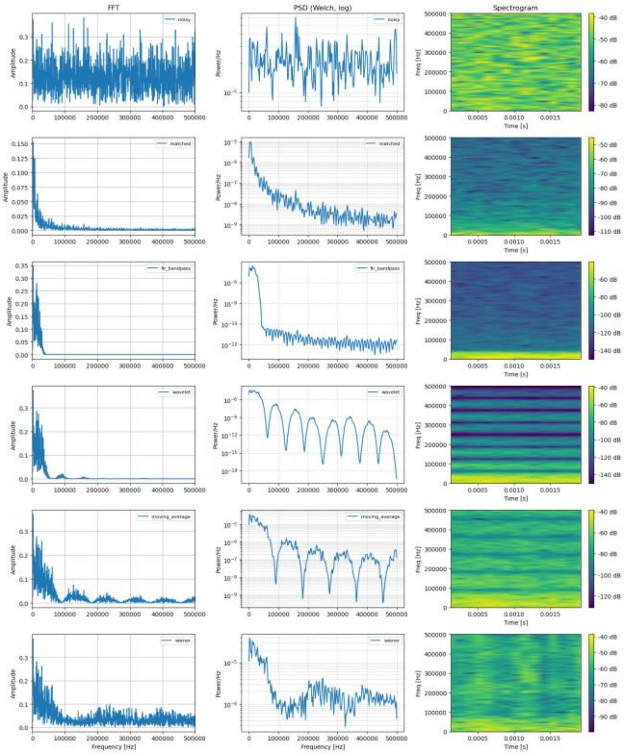


Fig. 33. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for λ = 10 poisson noise



Under Poisson Noise, the observed performance follows theoretical expectations:

Filter	Performance Reason
Matched Filter	Good peak enhancement because Poisson approximates Gaussian at moderate rate
Wiener Filter	Performs reasonably if Poisson approximates Gaussian
Wavelet	Effective due to spare-signal and non-Gaussian adaptivity
Fir Band-Pass	Removes deterministic drift but does not specifically target Poisson shot noise
Moving Averrage	Smooths signal but severaly degrades pulse precision

Thus, **Wavelet** remains most robust for Poisson shot-noise environments; matched filter close behind due to statistical convergence of Poisson to Gaussian at moderate intensity.

<u>Salt-and-pepper impulse noise was applied with corruption densities d ∈ {0.005, 0.01, 0.02}, representing the fraction of randomly flipped samples to minimum or maximum valu</u>

L. Salt & Pepper Noise with d = 0.005

Fig. 34. the original clean pulse and its noisy versions under d = 0.005 salt & pepper noise

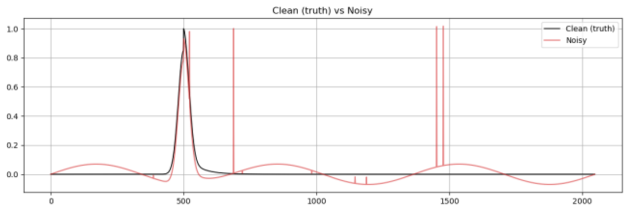


TABLE XII. FILTER PERFORMANCE METRICS FOR D = 0.005 SALT & PEPPER NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*4.0189e-03*	*23.959*	*5.771*	*0.867*	*0.0000*
wavelet	*4.0189e-03*	*23.959*	*5.771*	*0.867*	*0.0000*
moving average	*2.6639e-03*	*25.745*	*7.557*	*0.907*	*0.0000*
wiener	*3.9529e-03*	*24.031*	*5.843*	*0.868*	*0.0010*
matched	*1.9497e-02*	*17.100*	*-1.087*	*0.706*	*0.0000*
fir bandpass	*2.4510e-02*	*16.107*	*-2.081*	*0.121*	*0.0000*

Fig. 35. the time-domain outputs of all filters for d = 0.005 salt & pepper noise

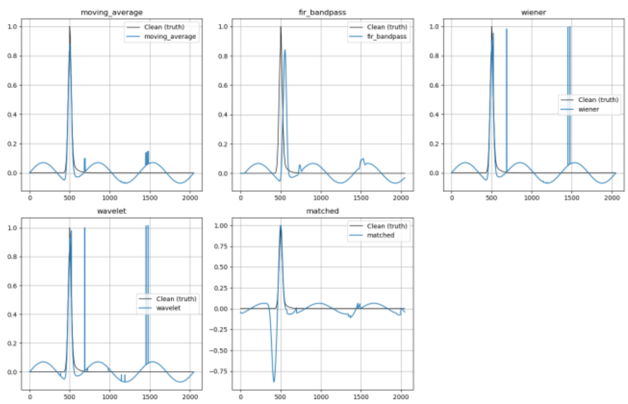
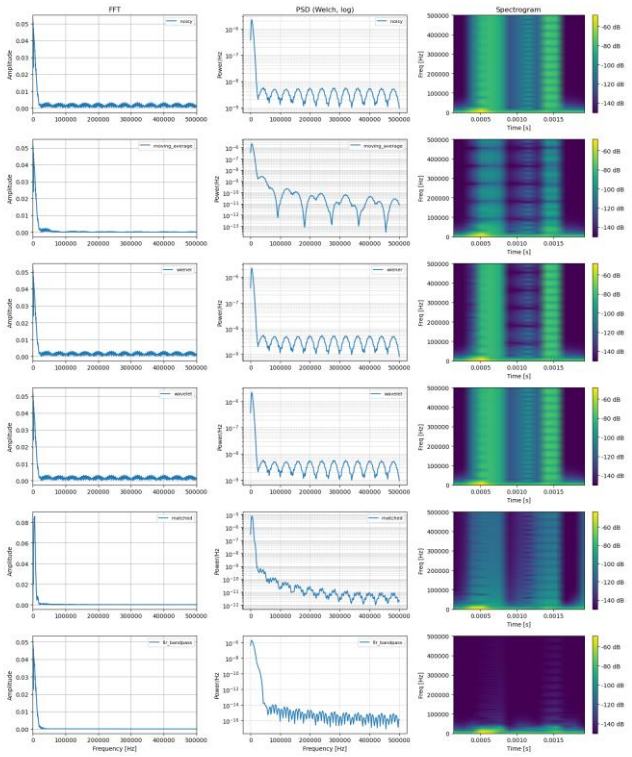


Fig. 36. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for d = 0.005 salt & pepper noise



M. Salt & Pepper Noise with d = 0.01

Fig. 37. the original clean pulse and its noisy versions under d = 0.01 salt & pepper noise

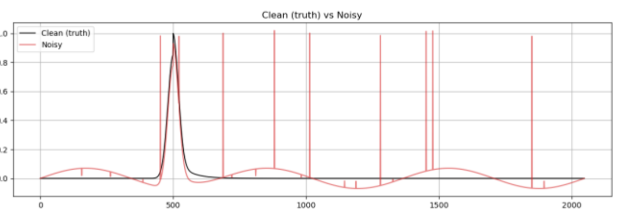


TABLE XIII. FILTER PERFORMANCE METRICS FOR D = 0.01 SALT & PEPPER NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*6.3727e-03*	*21.957*	*3.769*	*0.807*	*0.0000*
wavelet	*6.3727e-03*	*21.957*	*3.769*	*0.807*	*0.0010*
moving average	*2.7720e-03*	*25.572*	*7.384*	*0.904*	*0.0010*
wiener	*6.0302e-03*	*22.197*	*4.009*	*0.814*	*0.0000*
matched	*1.9353e-02*	*17.133*	*-1.055*	*0.708*	*0.0011*
fir bandpass	*2.3647e-02*	*16.262*	*-1.926*	*0.150*	*0.0000*

Fig. 38. the time-domain outputs of all filters for d = 0.01 salt & pepper noise

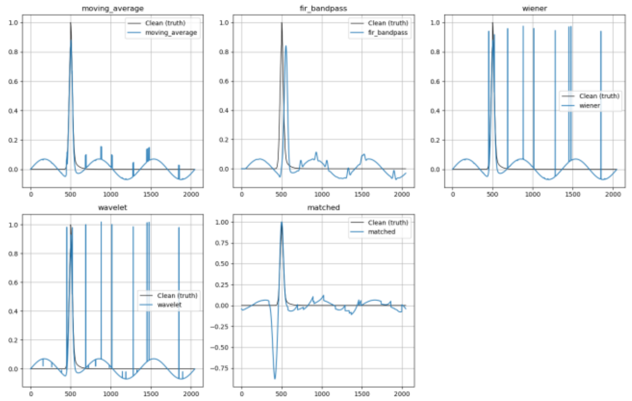
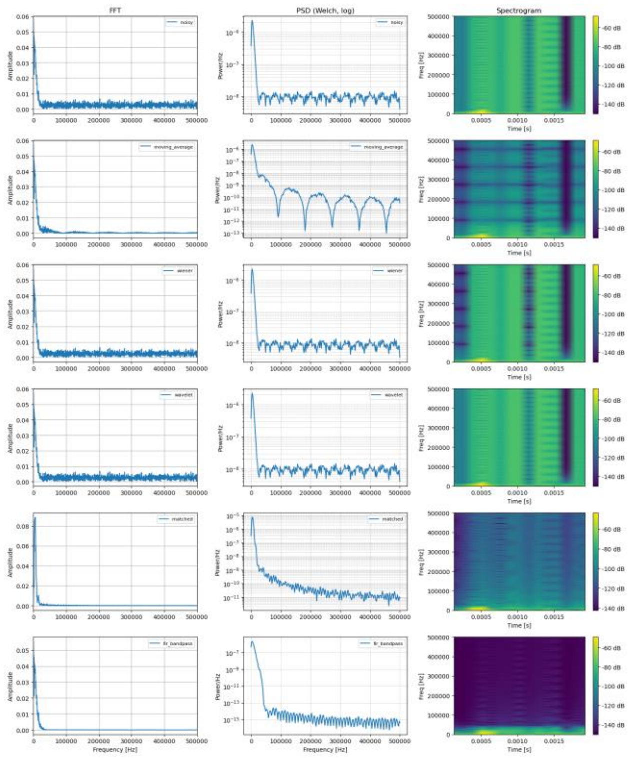


Fig. 39. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for d = 0.01 salt & pepper noise



N. Salt & Pepper Noise with d = 0.02

Fig. 40. the original clean pulse and its noisy versions under d = 0.02 salt & pepper noise

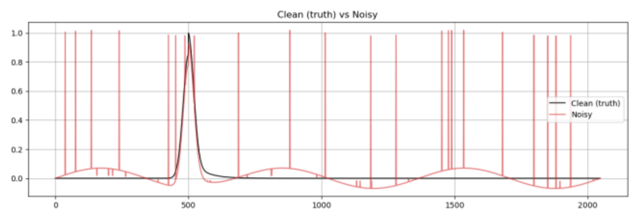


TABLE XIV. FILTER PERFORMANCE METRICS FOR D = 0.02 SALT & PEPPER NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*1.2257e-02*	*19.116*	*0.928*	*0.700*	*0.0000*
wavelet	*1.2257e-02*	*19.116*	*0.928*	*0.700*	*0.0010*
moving average	*3.2880e-03*	*24.831*	*6.643*	*0.890*	*0.0010*
wiener	*1.0291e-02*	*19.875*	*1.688*	*0.730*	*0.0000*
matched	*1.8528e-02*	*17.322*	*-0.866*	*0.713*	*0.0000*
fir bandpass	*2.3667e-02*	*16.259*	*-1.929*	*0.155*	*0.0010*

Fig. 41. the time-domain outputs of all filters for d = 0.02 salt & pepper noise

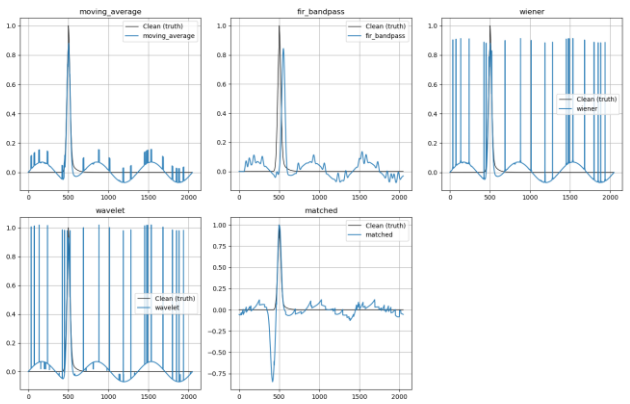
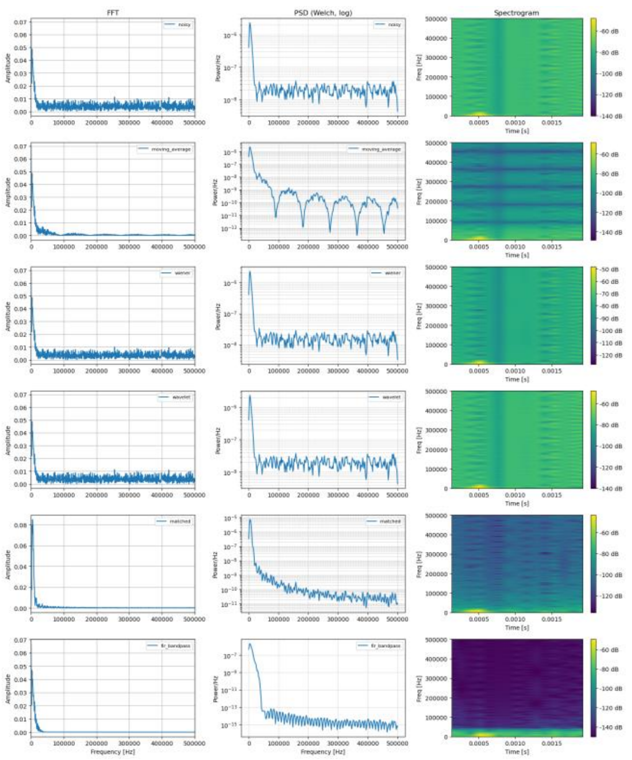


Fig. 42. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for d = 0.02 salt & pepper noise



Under Salt & Pepper Noise, the observed performance follows theoretical expectations:

Filter	Performance Reason
Matched Filter	Highly sensitive to impulsive spikes
Wiener Filter	Performs poorly because salt-pepper violates Gaussian noise assumption
Wavelet	Reduces impulse noise better than linear filters
Fir Band-Pass	Minimal improvement
Moving Averrage	Slight reduction but still bad for impulse noise

For **salt & pepper noise**, classical linear denoisers are insufficient; a median filter is typically required for optimal restoration. Matching filters perform poorly due to excessive amplification of impulse artifacts.

O. Gaussian Noise with σ = 0.1 and Poisson Noise with λ = 3

Fig. 43. the original clean pulse and its noisy versions under σ = 0.1 gaussian noise and λ = *3* poisson noise

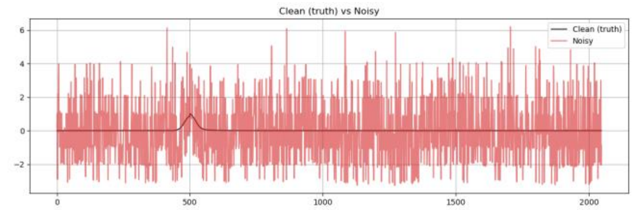


TABLE XV. FILTER PERFORMANCE METRICS FOR Σ = 0.1 GAUSSIAN NOISE AND Λ = *3* POISSON NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*2.8581e+00*	*-4.561*	*-22.748*	*0.075*	*0.0000*
wavelet	*2.2308e-01*	*6.515*	*-11.672*	*0.264*	*0.0012*
moving average	*2.9288e-01*	*5.333*	*-12.855*	*0.229*	*0.0000*
wiener	*4.6001e-01*	*3.372*	*-14.815*	*0.184*	*0.0010*
matched	*1.5684e-01*	*8.045*	*-10.142*	*-0.279*	*0.0005*
fir bandpass	*1.9577e-01*	*7.083*	*-11.105*	*-0.077*	*0.0000*

Fig. 44. the time-domain outputs of all filters for σ = 0.1 gaussian noise and λ = 3 poisson noise

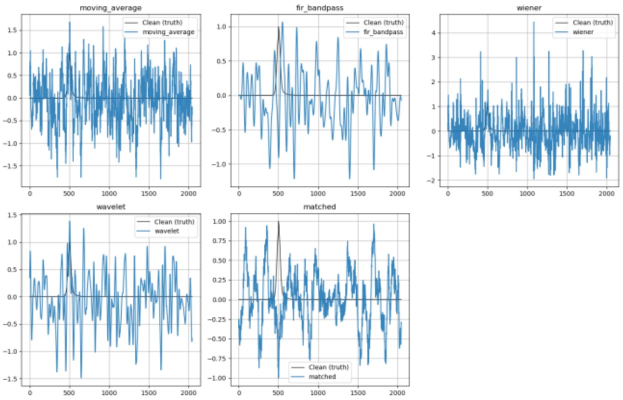
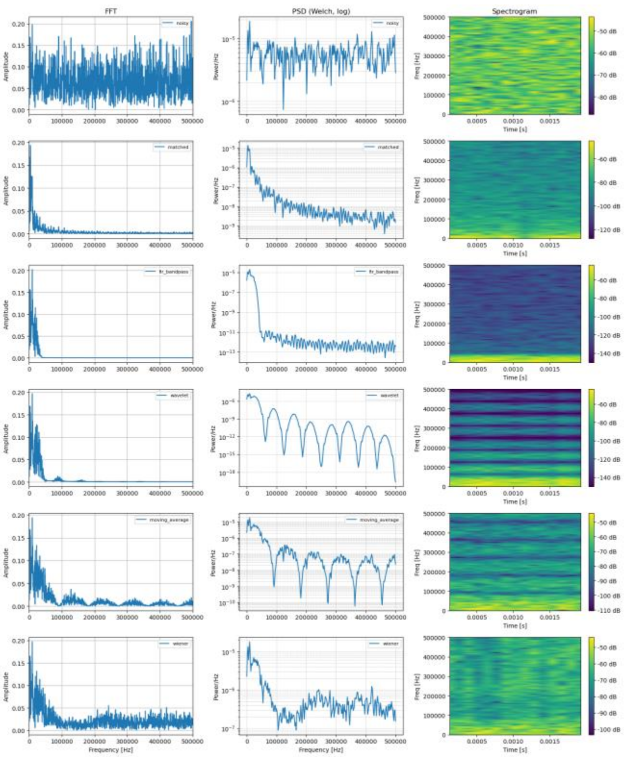


Fig. 45. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for σ = 0.1 gaussian noise and λ = 3 poisson noise



P. Gaussian Noise with σ = 0.1 and Salt & Pepper Noise with d = 0.03

Fig. 46. the original clean pulse and its noisy versions under σ = 0.1 gaussian noise and d = 0.03 salt & pepper noise

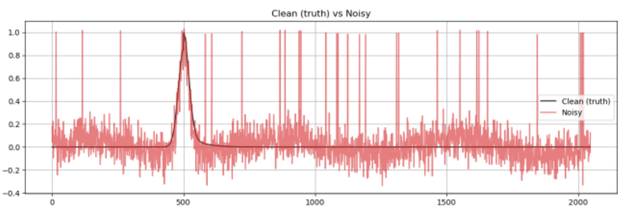


TABLE XVI. FILTER PERFORMANCE METRICS FOR Σ = 0.1 GAUSSIAN NOISE AND D = 0.03 SALT & PEPPER NOISE

Filter	MSE	PSNR (dB)	SNR (dB)	Correlation	Time (s)
noisy	*2.5025e-02*	*16.016*	*-2.171*	*0.554*	*0.0000*
wavelet	*5.2292e-03*	*22.816*	*4.628*	*0.833*	*0.0020*
moving average	*4.7781e-03*	*23.207*	*5.020*	*0.844*	*0.0000*
wiener	*1.2502e-02*	*19.030*	*0.842*	*0.686*	*0.0000*
matched	*1.9919e-02*	*17.007*	*-1.180*	*0.687*	*0.0000*
fir bandpass	*2.5591e-02*	*15.919*	*-2.269*	*0.095*	*0.0010*

Fig. 47. the time-domain outputs of all filters for σ = 0.1 gaussian noise and d = 0.03 salt & pepper noise

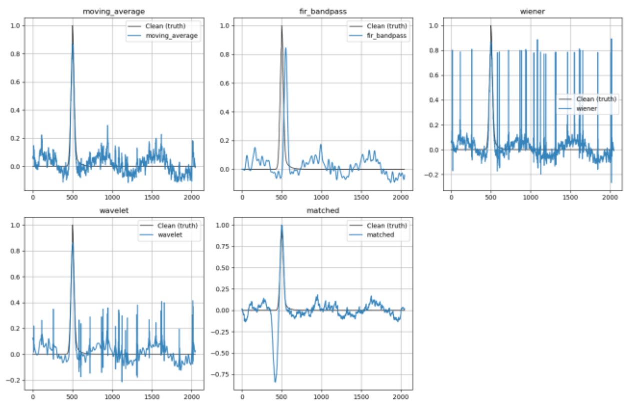
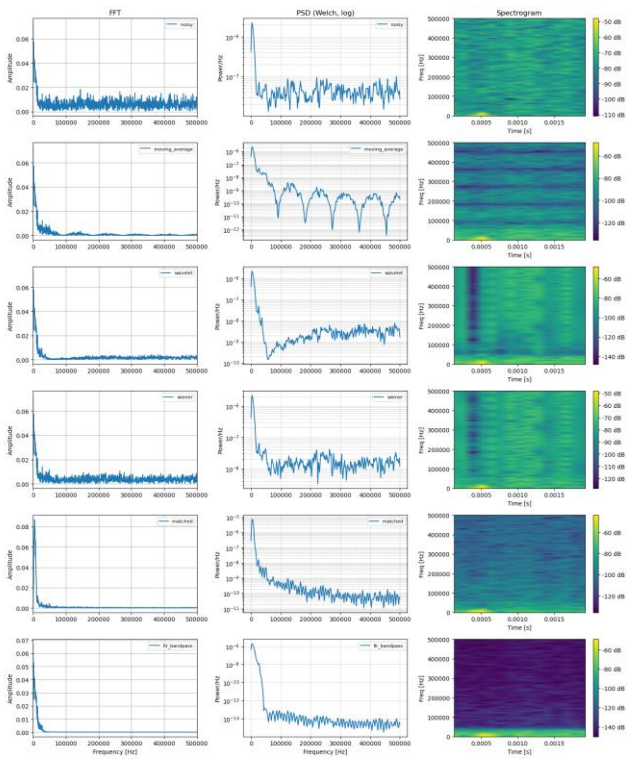


Fig. 48. the frequency-domain analysis using FFT, Power Spectral Density (PSD) and the time–frequency spectrograms for σ = 0.1 gaussian noise and d = 0.03 salt & pepper noise



VII. CONCLUSION

This study presented a comprehensive evaluation of classical digital filtering techniques for noisy pulse signal restoration under multiple noise environments, including Gaussian, uniform, Poisson, and salt-and-pepper disturbances. The results demonstrated that no single filter is universally optimal; instead, filtering performance is strongly dependent on the underlying noise characteristics.

Wavelet denoising consistently achieved the best overall performance across most noise types by adaptively suppressing noise while preserving sharp pulse transitions. The Wiener filter provided competitive results in Gaussian-like environments due to its local variance estimation, though it exhibited performance degradation under non-Gaussian impulsive noise. The matched filter excelled in scenarios dominated by Gaussian and Poisson noise, where the statistical structure aligned with its correlation-based design, but it performed poorly when faced with impulsive noise sources. FIR band-pass filtering proved effective for structured spectral noise and baseline drift removal, while the moving-average filter, although computationally simple, resulted in significant pulse distortion and was the least appropriate for precision-sensitive signals.

Overall, these findings highlight the importance of noise-aware filter selection in digital signal processing pipelines. For robust restoration across heterogeneous noise environments, wavelet-based denoising is the most reliable technique. Future work will extend this study toward adaptive hybrid systems capable of automatically classifying noise types and selecting the optimal filtering strategy in real time, enhancing signal quality in practical communication and sensing applications.

VIII. REFERANCES

[1] R. G. Lyons, Understanding Digital Signal Processing, 3rd ed. Pearson, 2011.

[2] S. K. Mitra, Digital Signal Processing: A Computer-Based Approach, 4th ed. McGraw-Hill, 2011.

[3] P. S. Reddy, "Implementation of Moving Average and FIR Filters for Noise Reduction in Signals," International Journal of Engineering Research & Technology, vol. 7, no. 5, pp. 1–5, 2018.

[4] SciPy Library Documentation — Signal Processing Module. (Accessed 2025). https://docs.scipy.org/doc/scipy/reference/signal.html

[5] PyWavelets Documentation — Wavelet Denoising Tools. (Accessed 2025). https://pywavelets.readthedocs.io/

[6] NumPy Documentation — Numerical Python. (Accessed 2025). https://numpy.org/