PROJECT REPORT

TITLE:

Digital Low-Pass Filtering Using Moving Average in Python

Introduction

In real-world systems, signals such as audio, sensor data, or communication waveforms often get corrupted by noise. Signal processing techniques are essential to clean, analyse, and extract meaningful information from such noisy data. One fundamental operation in digital signal processing (DSP) is filtering — used to remove unwanted components or features. This project focuses on implementing a simple yet effective moving average low-pass filter using Python to smooth a noisy sine wave and demonstrate noise reduction.

Objective

The main objective of this project is to simulate a noisy signal and apply a moving average low-pass filter to reduce the noise. The goal is to understand the basics of digital filtering and demonstrate its effectiveness using Python and its scientific libraries.

Methodology

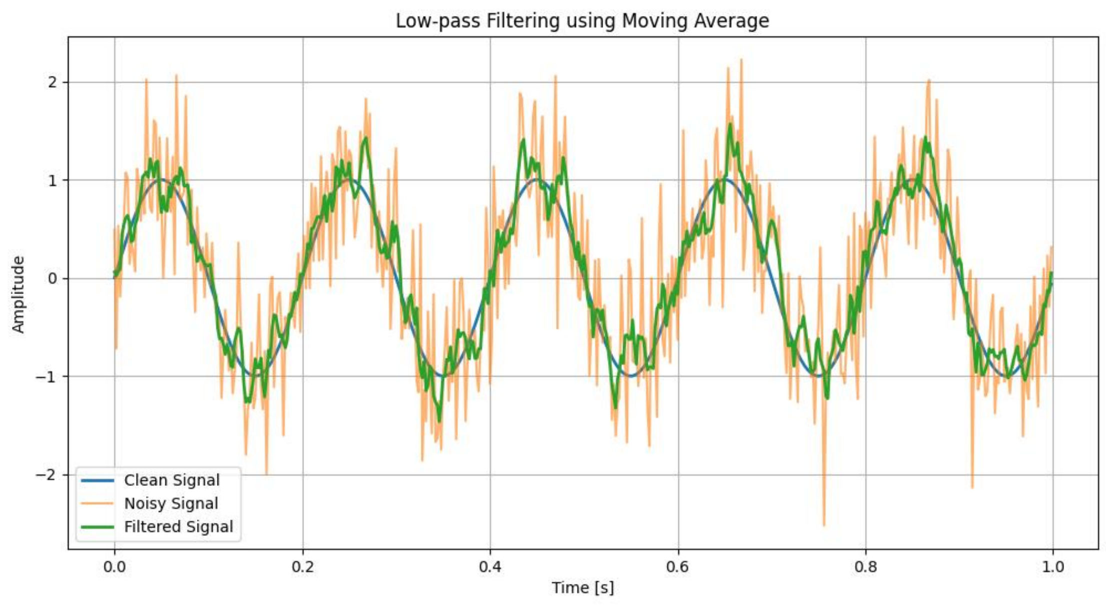
- A clean sine wave was generated using NumPy by defining a frequency of 5 Hz and sampling it at 500 Hz for 1 second.
- Gaussian noise (mean 0, standard deviation 0.5) was added to the clean signal to simulate real-world noise.
- A moving average filter was implemented using a convolution operation. A window of a fixed size (10 samples) slides over the signal, and each sample is replaced by the average of that window.
- The Python libraries used include:
    - NumPy for signal generation and filtering
    - Matplotlib for visualization

Result

The output of the project is a plot comparing:

- The original clean sine wave
- The noisy signal
- The filtered signal after applying the moving average

The filtered signal appears much smoother than the noisy one and closely follows the original clean waveform. Increasing the window size of the filter increases smoothness but can cause the signal to lag or lose detail. A balance is required between noise suppression and signal preservation.



Conclusion

This project successfully demonstrates the concept of digital low-pass filtering using a moving average method. The filter effectively reduced noise from the signal and improved its clarity. Through this project, the basics of signal generation, noise simulation, and filtering were practically understood. This knowledge forms a strong foundation for exploring more advanced DSP techniques in the future, such as FIR and IIR filters or using real-world sensor data.

References

- NumPy documentation: https://numpy.org/doc/
- Matplotlib documentation: https://matplotlib.org/stable/contents.html
- ChatGPT and online DSP tutorials for theoretical guidance
- Kaggle and other forums for signal processing tips and examples

Appendix

Python Code for Moving Average Filter:

```
import numpy as np

import matplotlib.pyplot as plt


fs = 500

t = np.arange(0, 1, 1/fs)

f = 5


clean_signal = np.sin(2 * np.pi * f * t)
```

```
noise = np.random.normal(0, 0.5, clean_signal.shape)
noisy_signal = clean_signal + noise


def moving_average(signal, window_size):
    return np.convolve(signal, np.ones(window_size)/window_size, mode='same')


window_size = 10
filtered_signal = moving_average(noisy_signal, window_size)


plt.figure(figsize=(12,6))
plt.plot(t, clean_signal, label='Clean Signal', linewidth=2)
plt.plot(t, noisy_signal, label='Noisy Signal', alpha=0.6)
plt.plot(t, filtered_signal, label='Filtered Signal', linewidth=2)
plt.xlabel('Time [s]')
plt.ylabel('Amplitude')
plt.title('Low-pass Filtering using Moving Average')
plt.legend()
plt.grid(True)
plt.savefig("dsp_filtered_graph.png", dpi=300)
plt.show()
```