Digital Information Processing Lab 2025

Class 4: Filters

Group 8 : Vision Matrix

17.06.2025

Name	Mat. Nr.	Email
Navya Sajeev Warrier	252782	navya.sajeev@st.ovgu.de
Sanjay Pulparambil Grish	249360	sanjay.pulparambil@st.ovgu.de

Class 4: Filters

Task: Filters

Get Acquainted:

- Familiarize yourself with the various filters and their parameters.

Choose Filters for ECG Signal:

- Select three filters to remove the three types of errors in the ECG signal *ecg_signal_all*.
- Explain which filter is suitable for each type of error and why.
- Set the filter parameters and demonstrate the before and after effects.
- Consider the order in which the filters should be applied.

Visualize the Filtering Process:

- Using the provided plot example, illustrate the filtering process of the ECG signal.
- Write a summary about the process

Introduction: Parts of the ECG

[ ] ↳ 3 cells hidden

Introduction of Error Types

[ ] ↳ 9 cells hidden

Filters

Here are some implementations of filters:

High-pass filter

[ ] ↳ 3 cells hidden

Notch Filter

[ ] ↳ 4 cells hidden

Wavelet-Denoising

Wavelet-Denoising can be used to remove high-frequency noise such as Gaussian noise while preserving important features of the signal.

[ ] ↳ 3 cells hidden

Median Filter

A median filter can be used to remove impulse noise while preserving the edges of the signal.

[ ] ↳ 3 cells hidden

Moving Average Filter

A moving average filter can be used to smooth high-frequency noise by averaging over a window.

[ ] ↳ 3 cells hidden

Fourier Transform Filter

A Fourier transform filter can be used to smooth high-frequency noise by transforming the signal into the frequency domain, removing unwanted frequencies, and transforming the signal back into the time domain.

[ ] ↳ 3 cells hidden

Savitzky-Golay Filter

A Savitzky-Golay filter can be used to smooth high-frequency noise while preserving important features of the signal. This filter is particularly good at preserving the signal integrity as it fits a polynomial to the data points and smooths them accordingly.

[ ] ↳ 3 cells hidden

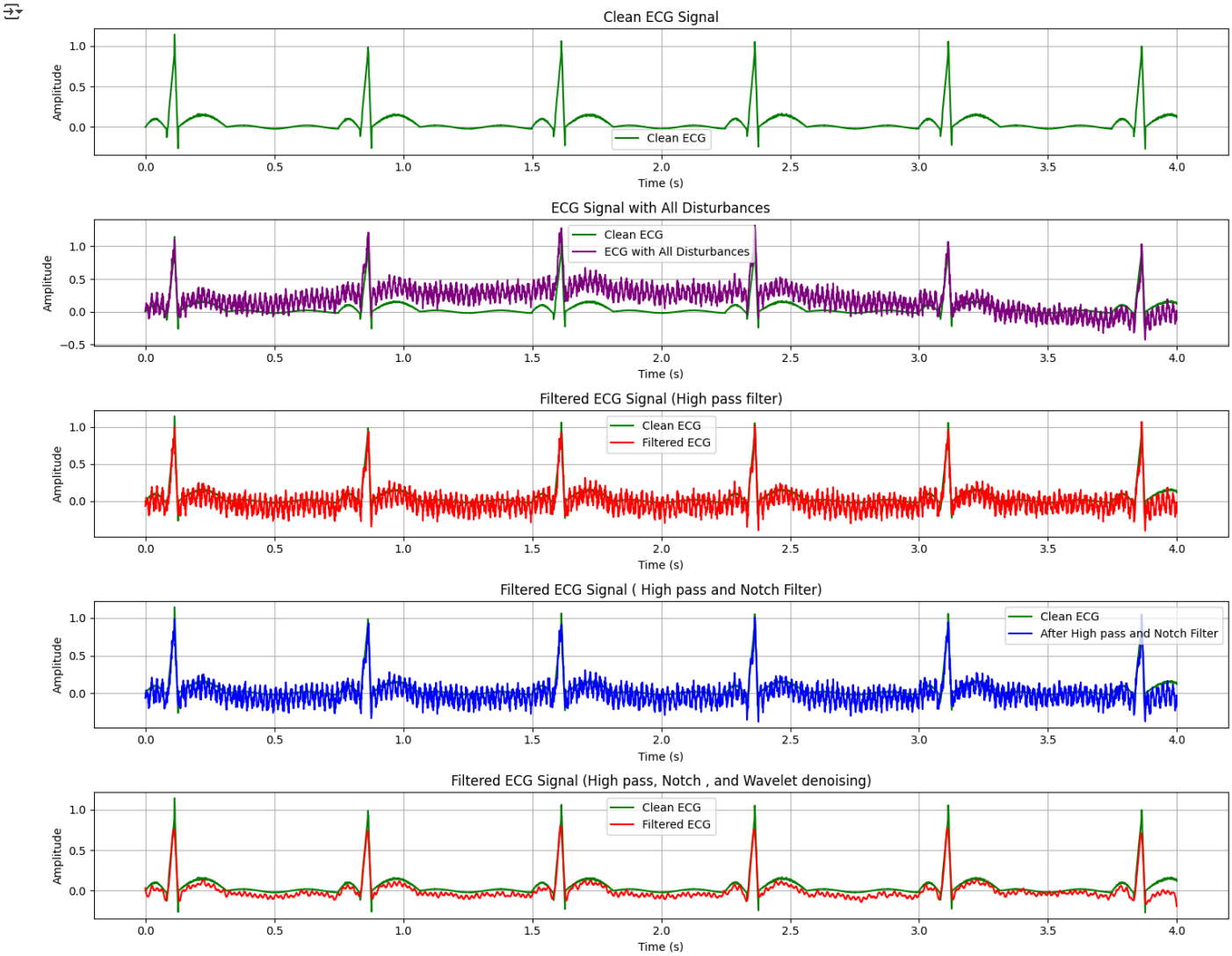
Adaptive Filtering with LMS

Adaptive filters continuously adjust to the characteristics of the input signal. The Least Mean Squares (LMS) algorithm is a commonly used adaptive filter, specifically designed for situations where the signal characteristics vary.

[ ] ↳ 3 cells hidden

Combination of Filters (Plot Example)

```
# Apply XXX - High-pass filter for baseline wander
filteredsig1 = apply_highpass_filter(ecg_signal_all, cutoff=0.5, fs=1000, order=5)

# Apply YYY - Notch filter for 50 Hz power line noise
filteredsig2 = apply_notch_filter(filteredsig1, cutoff=50, fs=1000, quality_factor=30)

# to compare and check Savitzky-Golay for Gaussian noise
filteredsig3 =  apply_savgol_filter(filteredsig2, 31, 3)

# Plot the signals
plt.figure(figsize=(15, 12))

plt.subplot(5, 1, 1)
plt.plot(t, ecg_signal, label='Clean ECG', color='green')
plt.title('Clean ECG Signal')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.legend()
plt.grid(True)

plt.subplot(5, 1, 2)
plt.plot(t, ecg_signal, label='Clean ECG', color='green')
plt.plot(t, ecg_signal_all, label='ECG with All Disturbances', color='purple')
plt.title('ECG Signal with All Disturbances')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.legend()
plt.grid(True)

plt.subplot(5, 1, 3)
plt.plot(t, ecg_signal, label='Clean ECG', color='green')
plt.plot(t, filteredsig1, label='Filtered ECG', color='red')
plt.title('Filtered ECG Signal (High pass filter)')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.legend()
plt.grid(True)

plt.subplot(5, 1, 4)
plt.plot(t, ecg_signal, label='Clean ECG', color='green')
plt.plot(t, filteredsig2, label='After High pass and Notch Filter', color='blue')
plt.title('Filtered ECG Signal ( High pass and Notch Filter)')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.legend()
plt.grid(True)

plt.subplot(5, 1, 5)
plt.plot(t, ecg_signal, label='Clean ECG', color='green')
plt.plot(t, filteredsig3, label='Filtered ECG', color='red')
plt.title('Filtered ECG Signal (High pass, Notch , and Wavelet denoising)')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.legend()
plt.grid(True)



plt.tight_layout()
plt.show()
```



Choose Filters for ECG Signal:

Select three filters to remove the three types of errors in the ECG signal ecg_signal_all. Explain which filter is suitable for each type of error and why. Set the filter parameters and demonstrate the before and after effects. Consider the order in which the filters should be applied.

---

3 filters selected are :

- High pass filter
- Notch filter
- Wavelet denoising

Step 1: High-pass Filter for Baseline Wander [1]

Because baseline drift is a **slow movement** of the whole ECG signal. If we don't remove it first, it will **mess up all the next filters**, especially the ones that work based on frequency.

- **Process:** Use a high-pass filter to attenuate low-frequency components, preserving higher frequencies. Proper cutoff frequency, sampling frequency, and filter order ensure baseline wander is removed without significant distortion of the ECG signal.

Parameters used:

```
apply_highpass_filter(signal, cutoff=0.5, fs=1000, order=5)
```

- `cutoff = 0.5 Hz` : Anything slower than this is probably drift.
- `fs = 1000` : This is how many data points per second your signal has (sampling rate).
- `order = 5` : The order controls how sharply the filter separates unwanted low-frequency noise (like baseline drift) from the useful signal. (for order = 1 Very gentle cutoff, Drift not fully removed ; for order = 10 Very sharp (steep transition), May distort or damage the signal)

-> Before and after effects of filter:

Before:

1. ECG signal "floats" up and down

2. Baseline is not centered (low-frequency noise present)

After:

1. Baseline is flattened
2. Signal is now centered around zero
3. QRS and waveforms preserved

Step 2: Notch Filter for 50 Hz Power Line Interference

Why second?

Because the noise is at **exactly 50 Hz —** a known frequency **—** and it doesn't go away with high-pass or low-pass filters.

How does a notch filter help?

It works like a scalpel: it **cuts out one specific frequency** (here, 50 Hz) and leaves everything else alone.

Parameters used:

```
apply_notch_filter(signal, cutoff=50, fs=1000, quality_factor=30)
```

- `cutoff = 50`: The frequency to remove.
- `fs = 1000`: Again, sampling rate.
- `quality_factor = 30`: Controls how narrow the notch is. Higher = more precise. The Quality Factor (Q) in a notch filter is like the sharpness of a noise-removing scalpel:

  Low Q (5) = too much gets cut (may remove 40 - 45 Hz of signal, might lose parts of ECG.)

  High Q (100) = barely any cut, might not remove enough of the noise

Q = 30 is just right for removing 50 Hz powerline hum without hurting your ECG.

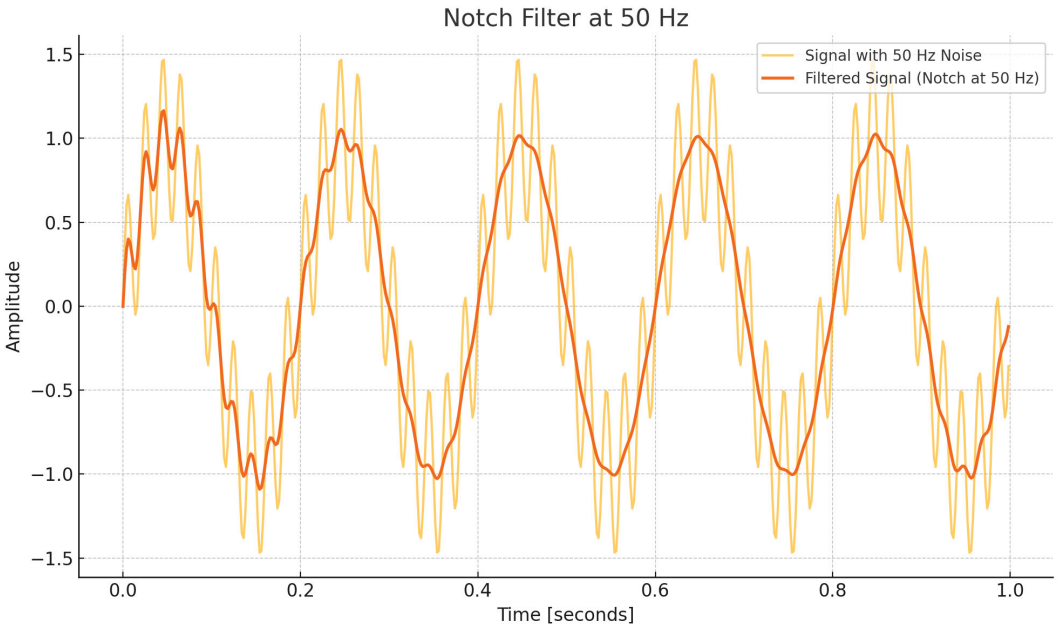
->Before and after effects of filter:

Before:

1. ECG has superimposed fine ripples or buzzing
2. Especially noticeable in flat regions between beats

After:

1. Ripples are eliminated
2. Clean waveform segments (no humming)
3. ECG features unaffected[4]



Certainly! Here's how you can reframe the explanation to focus on the **Savitzky-Golay filter** for Gaussian noise, following the same structure as your previous description:

Step 3: Savitzky-Golay Filter for Gaussian Noise[5]

Why last?

Process: The Savitzky-Golay filter smooths the signal by fitting a polynomial of a specified order to a moving window of data. This is done iteratively across the signal, allowing for noise reduction while preserving key features like peaks (e.g., the QRS complex in an ECG).

Because this noise is **Gaussian and typically high-frequency**, applying this filter last helps in smoothing out the residual noise without distorting important features of the signal. By doing so, we can retain the shape and key markers of the signal while reducing the overall noise.

```
dummy3 = apply_savgol_filter(dummy2, 31, 3)
```

By applying the Savitzky-Golay filter last, we:

1. **Stabilize the baseline** (remove drift),
2. **Remove high-frequency Gaussian noise** without damaging the underlying signal,
3. **Smooth the signal** while preserving key features, such as the P, QRS, and T waves in an ECG.

- `window_length=31` : This is the size of the moving window used to fit the polynomial. A larger window provides more smoothing but may blur sharp transitions (such as the QRS complex).
- `polyorder=3` : This parameter specifies the order of the polynomial. A cubic polynomial (order 3) balances noise reduction with preserving signal features. A lower polyorder (2 or 3) provides gentle smoothing without losing much detail. A higher polyorder (4 or above) could provide more fitting to the signal's local variations but might lead to unwanted distortions if the signal contains sharp transitions.

-> Before and After Effects of Filter:

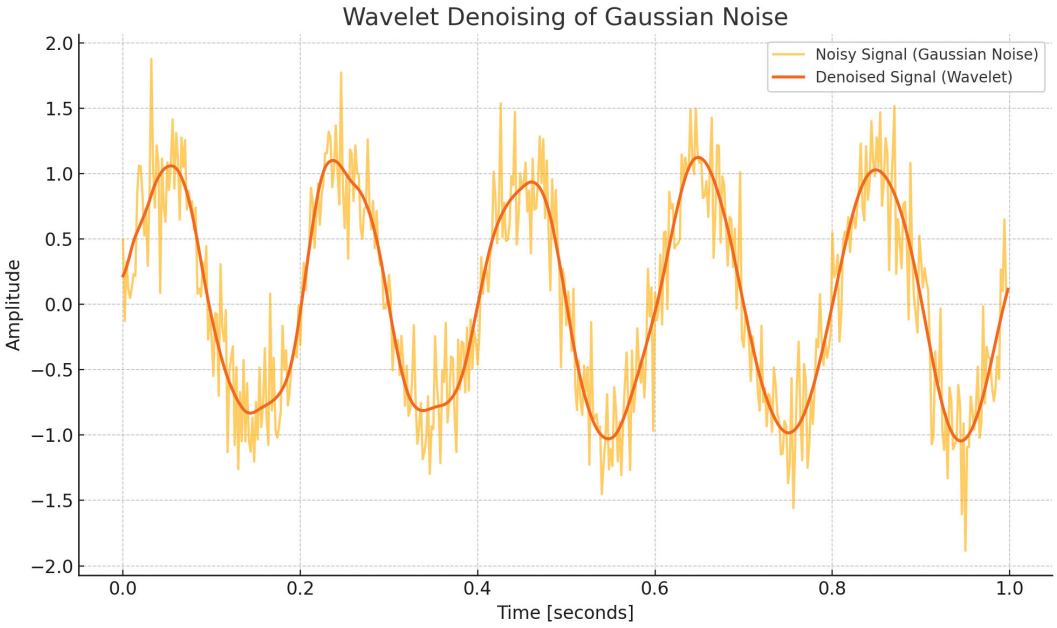
Before:

1. Signal exhibits high-frequency jitter and noise, especially in the P, QRS, and T waves.
2. Noise makes the peaks and valleys of the signal appear rough or irregular.

After:

1. The signal is smoothed, but the key waveforms (P, QRS, T) remain intact.
2. No distortion of peak positions or signal features.
3. Suitable for visualization or feature detection without excessive blurring.

This explanation emphasizes the benefits of the **Savitzky-Golay filter** and maintains clarity on its application and why it's beneficial to apply it as the last step in noise reduction.[2][3]



Summary about the process:

In this process, the noisy ECG signal ecg_signal_all was cleaned using a structured, three-step filtering pipeline. First, a high-pass filter was applied to remove baseline drift, which is a low-frequency shift caused by movement or respiration. Next, a notch filter targeted and eliminated 50 Hz power line interference without affecting nearby ECG frequencies. Finally, Savitzky-Golay was used to suppress high-frequency Gaussian noise while preserving the sharp features of the ECG waveform. Each filter addressed a specific noise type, and the order was carefully chosen to avoid interference between steps. The result is a clean, stable ECG signal suitable for analysis or diagnosis.

Reference

[1] ^ https://www.sciencedirect.com/science/article/pii/002207369290060D#:~:text=Abstract,linear%20and%20nonlinear%20filtering%20techniques.

[2] ^ Chatterjee, S., Thakur, R.S., Yadav, R.N., Gupta, L. and Raghuvanshi, D.K. (2020). Review of noise removal techniques in ECG signals. IET Signal Processing, [online] 14(9), pp.569–590. doi:https://doi.org/10.1049/iet-spr.2020.0104.

[3] ^ Zhu, Z., Gao, X., Cao, L., Pan, D., Cai, Y. and Zhu, Y. (2016). Analysis on the adaptive filter based on LMS algorithm. Optik, 127(11), pp.4698–4704. doi:https://doi.org/10.1016/j.ijleo.2016.02.005.

[4] Qureshi, R., Nawaz, M., Yar, F., Tunio, N. and Uzair, M. (2019). Analysis of ECG Signal Processing and Filtering Algorithms. International Journal of Advanced Computer Science and Applications, [online] 10(3). doi:https://doi.org/10.14569/ijacsa.2019.0100370.

[5] AlMahamdy, M. and Riley, H.B. (2014). Performance Study of Different Denoising Methods for ECG Signals. Procedia Computer Science, 37, pp.325–332. doi:https://doi.org/10.1016/j.procs.2014.08.048.