**PPG Signal Processing**

**1. Introduction**

Photoplethysmography (PPG) is a non-invasive technique used to measure blood volume changes in the microvascular tissue. This report presents the processing of a synthetic PPG signal, including filtering, peak and valley detection, heart rate estimation, and abnormality identification.

**2. Objectives**

* Generate a synthetic PPG signal.
* Apply a Butterworth filter to remove noise.
* Detect peaks and valleys of the PPG signal.
* Compute heart rate from detected peaks.
* Identify abnormal heart rate values.
* Visualize each processing step through plots.

**3. Methodology**

**3.1 PPG Signal Generation**

A synthetic PPG signal is generated using a sinusoidal function with added noise. The signal is sampled at 100 Hz over a duration of 10 seconds.

**3.2 Butterworth Filtering**

A low-pass Butterworth filter of order 3 with a cutoff frequency of 2 Hz is applied to remove high-frequency noise and preserve the main signal components.

**3.3 Peak and Valley Detection**

The find\_peaks function from the SciPy library is used to detect peaks (systolic points) and valleys (diastolic points) in the filtered PPG signal.

**3.4 Heart Rate Calculation**

Heart rate is estimated from the detected peaks using the time difference between consecutive peaks (RR intervals) and converting it to beats per minute (BPM).

**3.5 Abnormality Detection**

A heart rate below 50 BPM or above 100 BPM is considered abnormal. Detected abnormalities are displayed.

**Code:**

**import numpy as np**

**import matplotlib.pyplot as plt**

**from scipy.signal import butter, filtfilt, find\_peaks**

**# Generate synthetic PPG signal**

**def generate\_ppg\_signal(fs=100, duration=10):**

**t = np.linspace(0, duration, fs \* duration)**

**ppg\_signal = 1 + 0.5 \* np.sin(2 \* np.pi \* 1.2 \* t) + 0.2 \* np.random.randn(len(t))**

**return t, ppg\_signal**

**# Butterworth filter**

**def butterworth\_filter(signal, fs=100, cutoff=2, order=3):**

**nyquist = 0.5 \* fs**

**normal\_cutoff = cutoff / nyquist**

**b, a = butter(order, normal\_cutoff, btype='low', analog=False)**

**filtered\_signal = filtfilt(b, a, signal)**

**return filtered\_signal**

**# Detect peaks and valleys**

**def detect\_peaks\_and\_valleys(signal, distance=50):**

**peaks, \_ = find\_peaks(signal, distance=distance)**

**valleys, \_ = find\_peaks(-signal, distance=distance)**

**return peaks, valleys**

**# Calculate heart rate**

**def calculate\_heart\_rate(peaks, fs=100):**

**rr\_intervals = np.diff(peaks) / fs**

**heart\_rates = 60 / rr\_intervals**

**return heart\_rates**

**# Identify abnormalities**

**def detect\_abnormalities(heart\_rates):**

**abnormalities = []**

**for i, hr in enumerate(heart\_rates):**

**if hr < 50 or hr > 100:**

**abnormalities.append((i, hr))**

**return abnormalities**

**# Main execution**

**t, ppg\_signal = generate\_ppg\_signal()**

**filtered\_signal = butterworth\_filter(ppg\_signal)**

**peaks, valleys = detect\_peaks\_and\_valleys(filtered\_signal)**

**heart\_rates = calculate\_heart\_rate(peaks)**

**abnormalities = detect\_abnormalities(heart\_rates)**

**# Plot raw signal**

**plt.figure(figsize=(10, 4))**

**plt.plot(t, ppg\_signal, label='Raw PPG Signal', color='gray')**

**plt.title("Raw PPG Signal")**

**plt.xlabel("Time (s)")**

**plt.ylabel("Amplitude")**

**plt.legend()**

**plt.show()**

**# Plot filtered signal**

**plt.figure(figsize=(10, 4))**

**plt.plot(t, filtered\_signal, label='Filtered PPG Signal', color='green')**

**plt.title("Filtered PPG Signal")**

**plt.xlabel("Time (s)")**

**plt.ylabel("Amplitude")**

**plt.legend()**

**plt.show()**

**# Plot raw and filtered signal together**

**plt.figure(figsize=(10, 4))**

**plt.plot(t, ppg\_signal, label='Raw PPG Signal', alpha=0.7)**

**plt.plot(t, filtered\_signal, label='Filtered PPG Signal', linewidth=2)**

**plt.legend()**

**plt.title("Raw vs Filtered PPG Signal")**

**plt.xlabel("Time (s)")**

**plt.ylabel("Amplitude")**

**plt.show()**

**# Plot PPG signal with peaks and valleys**

**plt.figure(figsize=(10, 4))**

**plt.plot(t, filtered\_signal, label='Filtered PPG Signal')**

**plt.scatter(t[peaks], filtered\_signal[peaks], color='red', label='Peaks', marker='o')**

**plt.scatter(t[valleys], filtered\_signal[valleys], color='blue', label='Valleys', marker='x')**

**plt.legend()**

**plt.title("PPG Signal with Peaks and Valleys")**

**plt.xlabel("Time (s)")**

**plt.ylabel("Amplitude")**

**plt.show()**

**# Plot heart rate**

**plt.figure(figsize=(10, 4))**

**plt.plot(heart\_rates, marker='o', linestyle='-', label='Heart Rate')**

**plt.axhline(y=100, color='r', linestyle='--', label='High HR Threshold')**

**plt.axhline(y=50, color='b', linestyle='--', label='Low HR Threshold')**

**plt.legend()**

**plt.title("Heart Rate Over Time")**

**plt.xlabel("Beat Number")**

**plt.ylabel("Heart Rate (BPM)")**

**plt.show()**

**# Print abnormalities**

**if abnormalities:**

**print("Detected Abnormal Heart Rates:")**

**for index, hr in abnormalities:**

**print(f"Beat {index + 1}: {hr:.2f} BPM")**

**else:**

**print("No abnormal heart rates detected.")**

**4. Results and Analysis**

* **Raw and Filtered Signal Comparison:** The Butterworth filter effectively reduces noise while retaining the essential waveform characteristics.
* **Peak and Valley Detection:** The algorithm successfully identifies peaks and valleys in the signal.
* **Heart Rate Estimation:** The calculated heart rate is plotted to observe variations.
* **Abnormality Detection:** If any abnormal heart rate values are found, they are reported.

A screenshot of a computer screen

AI-generated content may be incorrect.

**5. Conclusion**

This lab successfully demonstrated PPG signal processing techniques, including noise filtering, peak and valley detection, heart rate estimation, and abnormality identification. The results highlight the importance of preprocessing for accurate physiological measurements.