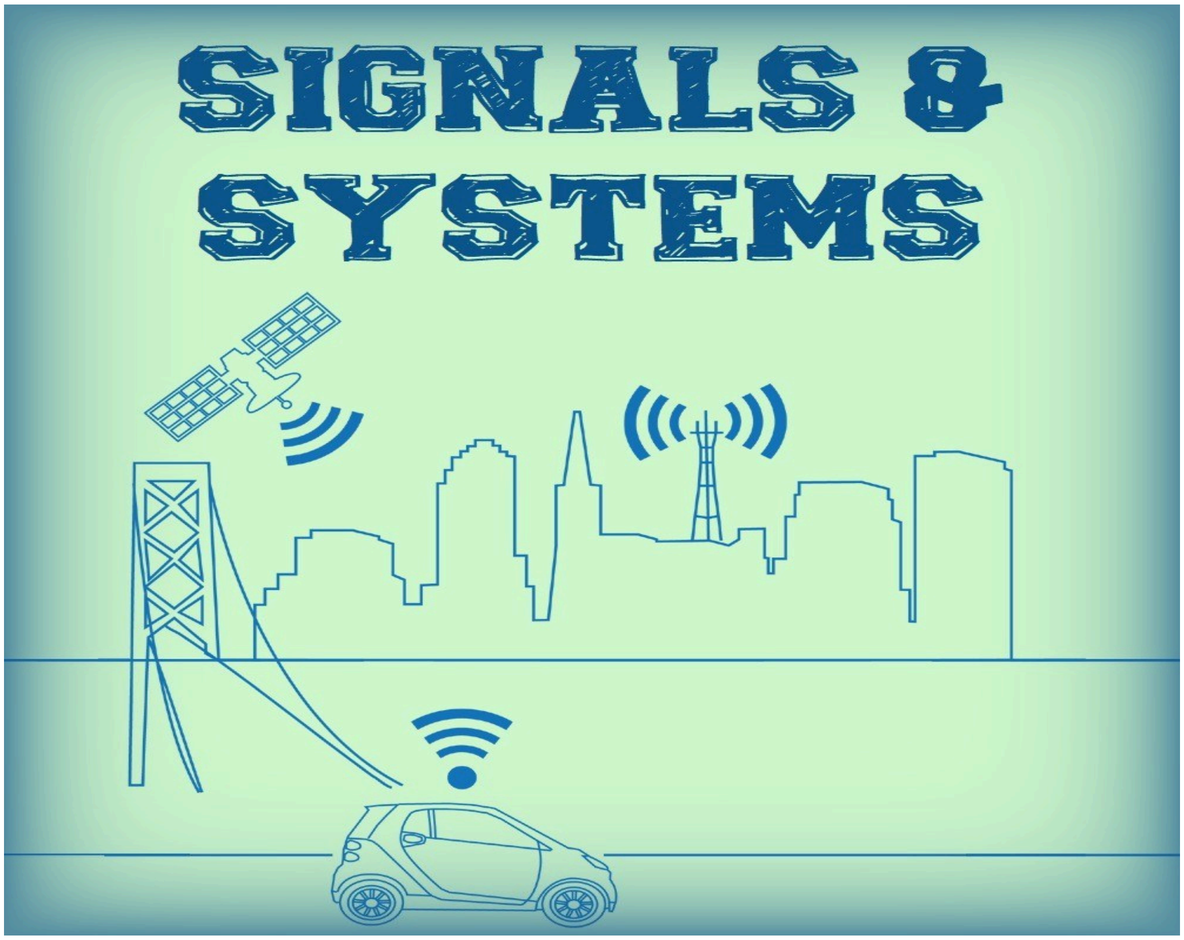
Programming assignment

Signals and Systems



PALAK BHAWSAR
(B22CH018)

Indian Institute of Technology , Jodhpur

Aim:

The objective of this project is to infer the type of filtering applied to an input signal based on the comparison between the filtered signals and the provided output signal. Three types of filters are considered: Low Pass, High Pass, and Band Pass. By implementing these filters, convolving them with the input signal, and correlating the convolved signals with the output signal, we aim to determine the filter type that best matches the provided output.

Background:

Filters are essential components in signal processing used to modify or extract specific features from signals. They can be broadly categorized into Low Pass, High Pass, and Band Pass filters based on their frequency response characteristics.

Types of Filters:

Low Pass Filter (LPF)

It is a type of filter commonly used in signal processing to allow signals with frequencies below a certain cutoff frequency to pass through while attenuating signals with frequencies above the cutoff frequency. LPFs are widely employed in various applications such as audio processing, telecommunications, biomedical engineering, and control systems.

Characteristics of a Low Pass Filter:

Frequency Response: The frequency response of an LPF describes how the filter behaves with respect to different frequencies in the input signal. It is characterized by allowing low-frequency components to pass through unaffected (i.e., with minimal attenuation) while progressively attenuating higher-frequency components.

Cutoff Frequency: The cutoff frequency of an LPF is the frequency at which the filter begins to attenuate the signal. Frequencies below the cutoff frequency are considered to be in the passband, while frequencies above the cutoff frequency are in the stopband.

Attenuation: LPFs provide increasing levels of attenuation for frequencies above the cutoff frequency. The rate at which the attenuation increases beyond the cutoff frequency is determined by the filter's design parameters, such as its order.

High Pass Filter (HPF) :

A High Pass Filter (HPF) is a type of filter used in signal processing to allow signals with frequencies above a certain cutoff frequency to pass through while attenuating signals with frequencies below the cutoff frequency. HPFs are employed in various applications where it is necessary to remove low-frequency components from a signal while preserving or enhancing high-frequency components.

Characteristics of a High Pass Filter:

Frequency Response: The frequency response of an HPF is characterized by allowing high-frequency components of the input signal to pass through unaffected (with minimal attenuation) while progressively attenuating lower-frequency components.

Cutoff Frequency: Similar to a low pass filter, the cutoff frequency of an HPF is the frequency at which the filter begins to attenuate the signal. Frequencies above the cutoff frequency are considered to be in the passband, while frequencies below the cutoff frequency are in the stopband.

Attenuation: HPFs provide increasing levels of attenuation for frequencies below the cutoff frequency. The rate at which the attenuation increases below the cutoff frequency is determined by the filter's design parameters, such as its order.

Band Pass Filter (BPF) :

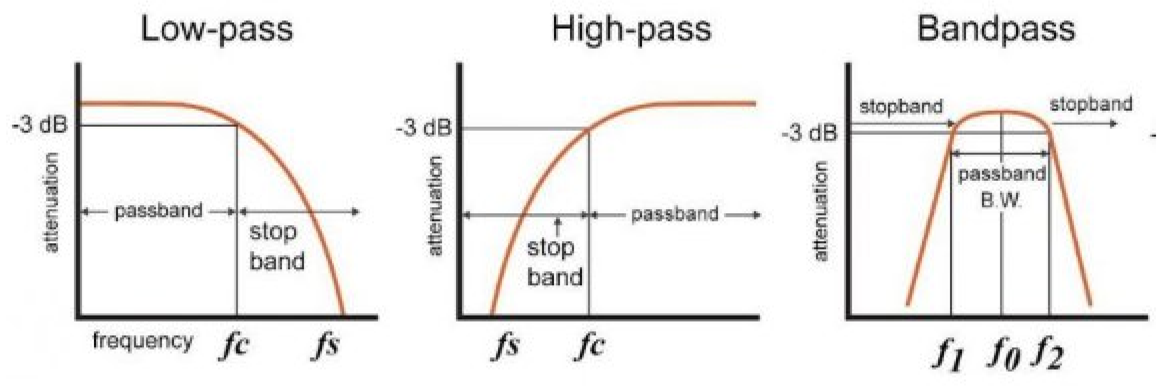
A Band Pass Filter (BPF) is a type of filter used in signal processing to allow signals within a specific frequency band to pass through while attenuating frequencies outside of this band. BPFs are valuable tools for extracting and isolating specific frequency components from a signal while rejecting unwanted frequencies.

Characteristics of a Band Pass Filter:

Frequency Response: The frequency response of a BPF is characterized by allowing a specific range of frequencies, defined by lower and upper cutoff frequencies, to pass through relatively unaffected, while attenuating frequencies outside of this range.

Cutoff Frequencies: BPFs have two cutoff frequencies: a lower cutoff frequency and an upper cutoff frequency. These frequencies define the bandwidth of the filter and determine the range of frequencies that are allowed to pass through.

Passband and Stopband: Frequencies within the range defined by the lower and upper cutoff frequencies are considered to be in the passband, while frequencies outside of this range are in the stopband.



Convolution:

Convolution is a fundamental mathematical operation employed in signal processing to describe the interaction between two signals as they overlap and slide relative to each other. It plays a pivotal role in various applications, including filtering, system analysis, and feature extraction.

Mathematical Definition:

In the context of discrete signals, the convolution operation between two signals x(t) and h(t):

$$(x * h)(t) = \int_{-\infty}^{\infty} x(\tau) \cdot h(t - \tau) \, d\tau$$

where:

- $(x * h)(t)$ represents the convolution of $x(t)$ and $h(t)$ at time t.
- $x(\tau)$ and $h(t - \tau)$ are the values of $x(t)$ and $h(t)$ at time t and τ, respectively.

In discrete form, the convolution equation simplifies to:

$$(x * h)(n) = \sum_{m=-\infty}^{\infty} x(m) \cdot h(n-m)$$

where:

- $(x * h)(n)$ denotes the convolution of $x(n)$ and $h(n)$ at index n.
- $x(m)$ and $h(n-m)$ are the values of $x(n)$ and $h(n)$ at indices m and $n-m$, respectively.

Interpretation:

- Convolution can be interpreted as a sliding dot product operation between the two signals. At each point in time (or index), the signals are multiplied element-wise, and the products are summed to yield the output value of the convolution at that point.
- In signal processing, convolution is commonly used for filtering operations. By convolving a signal with an impulse response (filter kernel), we effectively filter the signal to achieve desired frequency response characteristics.

Correlation:

Correlation is a statistical measure used to quantify the degree of association or similarity between two signals or datasets. In signal processing, correlation plays a crucial role in analyzing the relationship between signals, identifying patterns, and determining similarity metrics.

Types of Correlation:

Pearson Correlation Coefficient: Also known as the linear correlation coefficient, Pearson correlation measures the linear relationship between two variables. It ranges from -1 to 1, where:

- A correlation coefficient of 1 indicates a perfect positive linear relationship.
- A correlation coefficient of -1 indicates a perfect negative linear relationship.
- A correlation coefficient of 0 indicates no linear relationship.
-

Cross-Correlation: Cross-correlation measures the similarity between two signals as they shift relative to each other. It is commonly used in signal processing to analyze time-dependent relationships between signals. Cross-correlation is defined as follows:

$$(f \star g)(\tau) = \int_{-\infty}^{\infty} f(t) \cdot g(t+\tau)\, dt$$

Autocorrelation: Autocorrelation measures the similarity between a signal and a delayed version of itself. It is particularly useful in analyzing periodic or cyclical patterns within a signal. Autocorrelation is defined as:

$$R(\tau) = \int_{-\infty}^{\infty} f(t) \cdot f(t+\tau)\, dt$$

Interpretation:

- A high correlation coefficient indicates a strong positive correlation between the signals, meaning they tend to vary in the same direction.
- A low or negative correlation coefficient indicates a weak or negative correlation, suggesting an inverse relationship between the signals.
- Correlation coefficients close to zero imply little to no linear relationship between the signals.

Methodology:

1. Filter Design:

- **Butterworth Filter Implementation:** Leveraging the robust capabilities of the scipy.signal library, we embark on the design and implementation of three essential types of filters: Low Pass, High Pass, and Band Pass. The Butterworth filter design method is employed as it offers a balance between passband flatness and stopband attenuation.
- **Specification of Cutoff Frequencies and Filter Orders:** Each filter is meticulously configured based on specified cutoff frequencies and filter orders. These parameters play a pivotal role in shaping the frequency response characteristics of the filters.

2. Filtering:

- Application of Filters to Input Signal: With the filters now instantiated, we proceed to filter the input signal using each of the designed filters. This process yields three distinct filtered signals: ylp(t),yhp(t) and ybp(t) each embodying unique spectral properties dictated by the filtering operation.

3. Convolution:

- Mathematical Relationship Exploration: Building upon the filtered signals obtained in the previous step, we delve into the realm of convolution. This mathematical operation serves as a cornerstone in elucidating the relationship between the filtered signals and the provided output signal. By convolving each filtered signal with the output signal, we unveil critical insights into their congruence and divergence.

4. Correlation:

- Quantifying Signal Similarity: Correlation emerges as a pivotal tool in our analytical arsenal, enabling the quantification of similarity between signals. By calculating the correlation coefficients between the convolved signals and the output signal, we gain valuable insights into the degree of alignment between the filtered signals and the desired output.

5. Selection:

- Identification of Optimal Filter: Drawing upon the correlation coefficients obtained from the preceding analysis, we undertake the crucial task of identifying the filter type that manifests the highest degree of correlation with the provided output signal. This meticulous selection process culminates in the identification of the optimal filter type that best aligns with the desired output.

Data:

The input data consists of two text files:

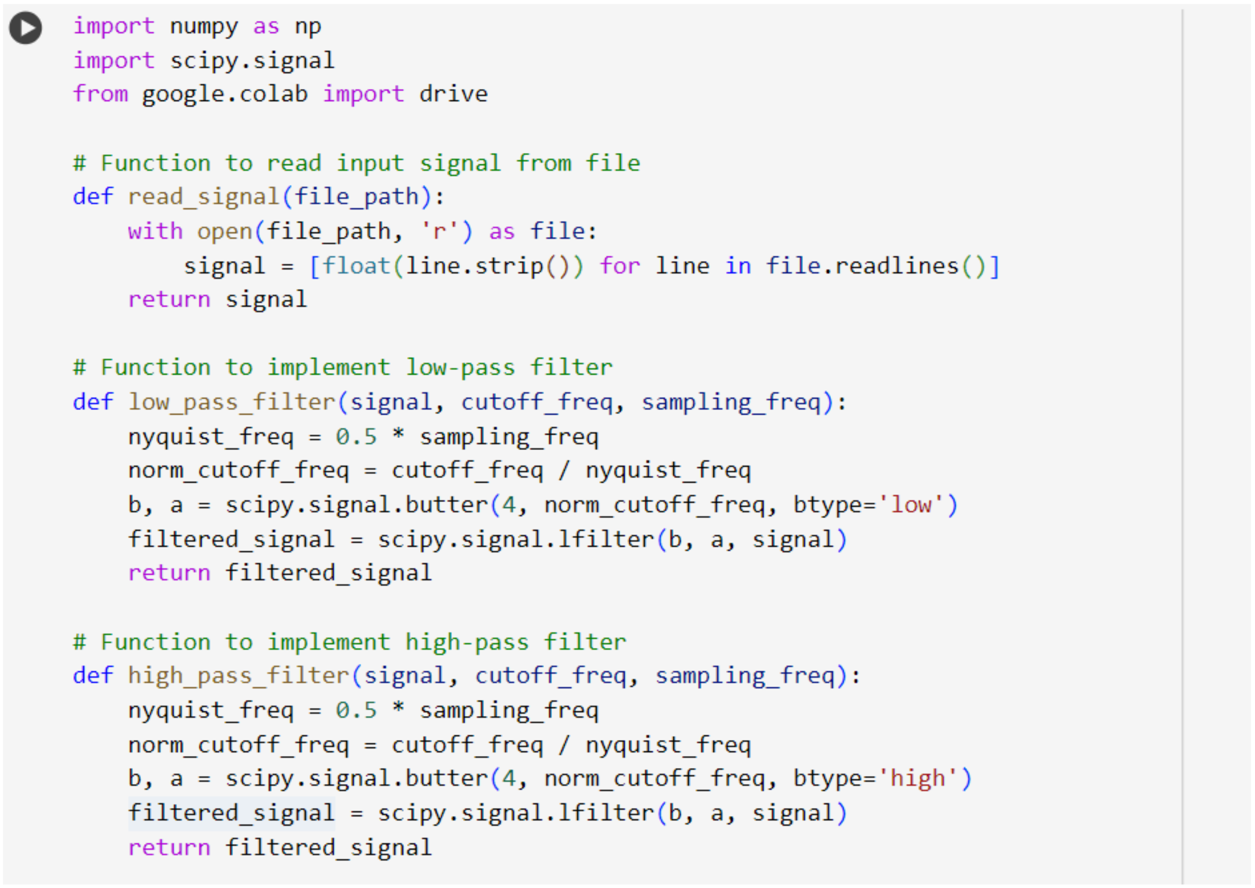
Input Signal (x(t)): Contains the raw input signal data.

```
-0.01
-0.01
-0.00
-0.00
-0.00
-0.00
-0.00
0.00
0.00
0.00
0.00
0.00
0.00
```

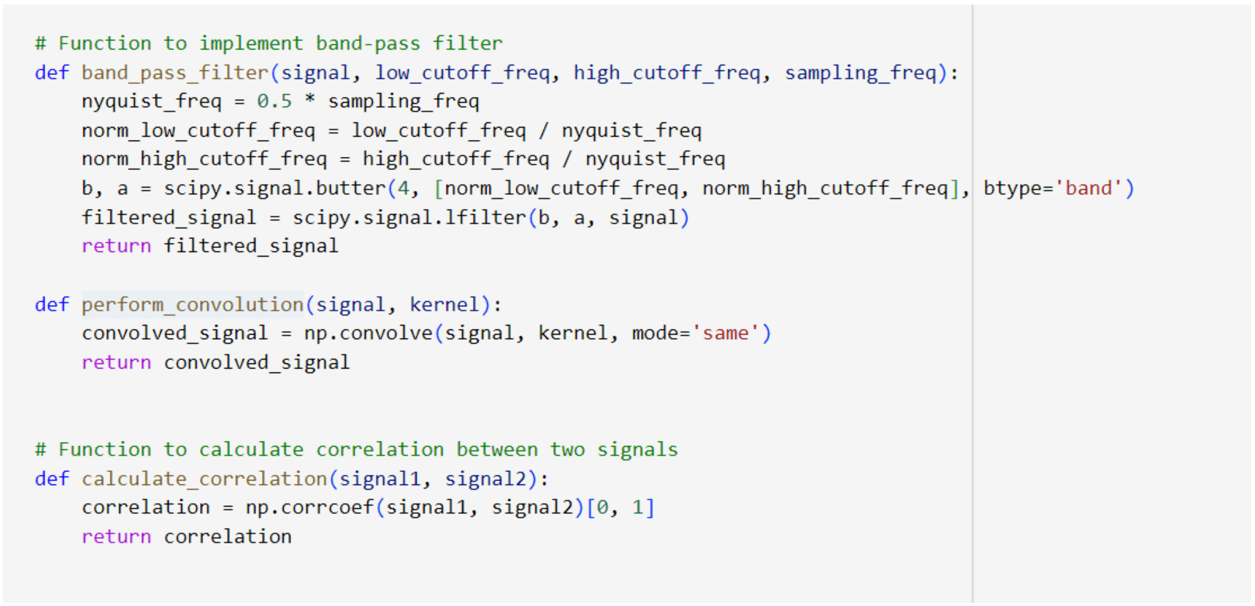
Output Signal (y(t)): Contains the filtered output signal data.

```
0.00
-0.00
-0.00
-0.00
-0.00
-0.00
-0.00
-0.00
-0.00
-0.00
-0.00
-0.00
```

Code:



```python
import numpy as np
import scipy.signal
from google.colab import drive

# Function to read input signal from file
def read_signal(file_path):
    with open(file_path, 'r') as file:
        signal = [float(line.strip()) for line in file.readlines()]
    return signal

# Function to implement low-pass filter
def low_pass_filter(signal, cutoff_freq, sampling_freq):
    nyquist_freq = 0.5 * sampling_freq
    norm_cutoff_freq = cutoff_freq / nyquist_freq
    b, a = scipy.signal.butter(4, norm_cutoff_freq, btype='low')
    filtered_signal = scipy.signal.lfilter(b, a, signal)
    return filtered_signal

# Function to implement high-pass filter
def high_pass_filter(signal, cutoff_freq, sampling_freq):
    nyquist_freq = 0.5 * sampling_freq
    norm_cutoff_freq = cutoff_freq / nyquist_freq
    b, a = scipy.signal.butter(4, norm_cutoff_freq, btype='high')
    filtered_signal = scipy.signal.lfilter(b, a, signal)
    return filtered_signal
```



```python
# Function to implement band-pass filter
def band_pass_filter(signal, low_cutoff_freq, high_cutoff_freq, sampling_freq):
    nyquist_freq = 0.5 * sampling_freq
    norm_low_cutoff_freq = low_cutoff_freq / nyquist_freq
    norm_high_cutoff_freq = high_cutoff_freq / nyquist_freq
    b, a = scipy.signal.butter(4, [norm_low_cutoff_freq, norm_high_cutoff_freq], btype='band')
    filtered_signal = scipy.signal.lfilter(b, a, signal)
    return filtered_signal

def perform_convolution(signal, kernel):
    convolved_signal = np.convolve(signal, kernel, mode='same')
    return convolved_signal


# Function to calculate correlation between two signals
def calculate_correlation(signal1, signal2):
    correlation = np.corrcoef(signal1, signal2)[0, 1]
    return correlation
```

```python
# File paths in your Google Drive
input_signal_path = '/content/INPUT-SIGNAL-X(t).txt'
output_signal_path = '/content/OUTPUT-SIGNAL-Y(t).txt'

# Load input and output signals
input_signal = read_signal(input_signal_path)
output_signal = read_signal(output_signal_path)

# Define the cutoff frequency and sampling frequency
cutoff_freq = 1000  # Replace with the actual cutoff frequency in Hz
sampling_freq = 5000  # Replace with the actual sampling frequency in Hz

# Filter input signal using low-pass filter
filtered_low_pass = low_pass_filter(input_signal, cutoff_freq, sampling_freq)

low_cutoff_freq = 100
high_cutoff_freq = 1000  # Replace with the actual high cutoff frequency in Hz

# Filter input signal using high-pass filter
filtered_high_pass = high_pass_filter(input_signal, cutoff_freq, sampling_freq)

# Filter input signal using band-pass filter
filtered_band_pass = band_pass_filter(input_signal, low_cutoff_freq, high_cutoff_freq, sampling_freq)

# Perform convolution between filtered signals and output signal
convolved_low_pass = perform_convolution(filtered_low_pass, output_signal)
convolved_high_pass = perform_convolution(filtered_high_pass, output_signal)
convolved_band_pass = perform_convolution(filtered_band_pass, output_signal)
```

```python
# Calculate correlation between filtered signals and output signal
correlation_low_pass = calculate_correlation(filtered_low_pass, output_signal)
correlation_high_pass = calculate_correlation(filtered_high_pass, output_signal)
correlation_band_pass = calculate_correlation(filtered_band_pass, output_signal)

# Take absolute values of correlations
correlation_low_pass = abs(correlation_low_pass)
correlation_high_pass = abs(correlation_high_pass)
correlation_band_pass = abs(correlation_band_pass)

best_filter_array = [correlation_low_pass, correlation_high_pass, correlation_band_pass]
best_filter = np.argmax(best_filter_array)

# Print the result
if best_filter == 0:
    print("The best match is the Low Pass Filter.")
    print("Correlation is: ", best_filter_array[best_filter])
elif best_filter == 1:
    print("The best match is the High Pass Filter.")
    print("Correlation is: ", best_filter_array[best_filter])
else:
    print("The best match is the Band Pass Filter.")
    print("Correlation is: ", best_filter_array[best_filter])
```

```python
import matplotlib.pyplot as plt

# Plot input signal, output signal, and filtered signals
plt.figure(figsize=(12, 8))

plt.subplot(3, 1, 1)
plt.plot(input_signal, color='blue', label='Input Signal (x(t))')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Input Signal')

plt.subplot(3, 1, 2)
plt.plot(output_signal, color='red', label='Output Signal (y(t))')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Output Signal')

plt.subplot(3, 1, 3)
plt.plot(filtered_low_pass, color='green', label='Filtered Low Pass Signal')
plt.plot(filtered_high_pass, color='orange', label='Filtered High Pass Signal')
plt.plot(filtered_band_pass, color='purple', label='Filtered Band Pass Signal')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Filtered Signals')
plt.legend()

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

```python
# Create a heatmap to visualize correlation coefficients
correlation_matrix = np.array([[correlation_low_pass, correlation_high_pass, correlation_band_pass]])
plt.figure(figsize=(8, 6))
plt.imshow(correlation_matrix, cmap='hot', interpolation='nearest')
plt.title('Correlation Coefficients')
plt.xlabel('Filters')
plt.ylabel('Correlation')
plt.colorbar()
plt.xticks(np.arange(3), ['Low Pass', 'High Pass', 'Band Pass'])
plt.yticks([0], ['Correlation'])
plt.show()
```

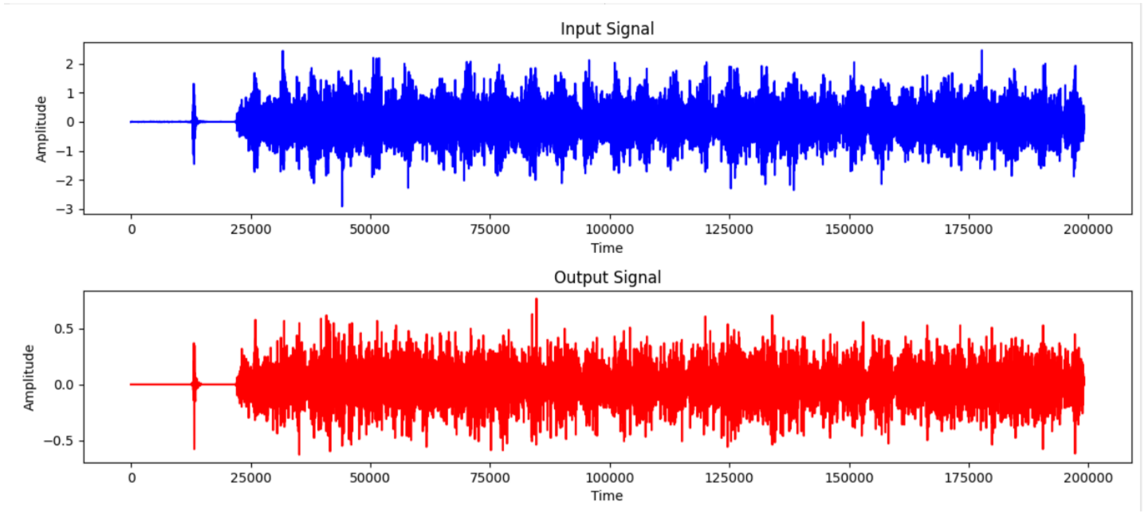
```python
# Plot filtered signals obtained through convolution
plt.figure(figsize=(12, 8))

plt.subplot(3, 1, 1)
plt.plot(convolved_low_pass, color='green', label='Convolved Low Pass Signal')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Convolved Low Pass Signal')

plt.subplot(3, 1, 2)
plt.plot(convolved_high_pass, color='orange', label='Convolved High Pass Signal')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Convolved High Pass Signal')

plt.subplot(3, 1, 3)
plt.plot(convolved_band_pass, color='purple', label='Convolved Band Pass Signal')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.title('Convolved Band Pass Signal')

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

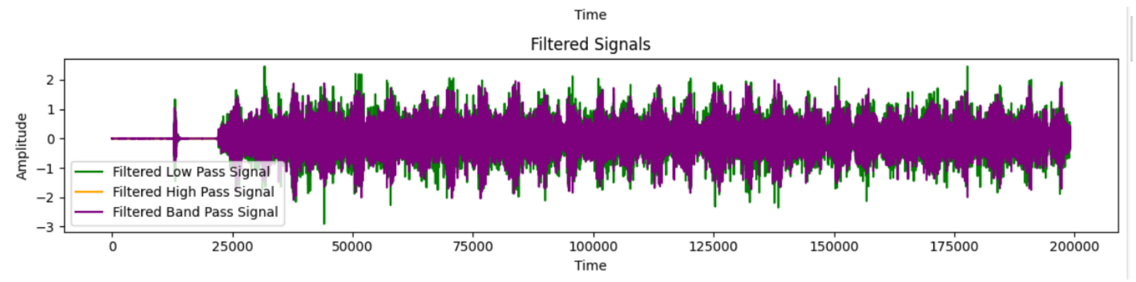
```python
import matplotlib.pyplot as plt

# Create a bar plot to visualize correlation coefficients
filters = ['Low Pass', 'High Pass', 'Band Pass']
correlation_values = [correlation_low_pass, correlation_high_pass, correlation_band_pass]

plt.figure(figsize=(8, 6))
plt.bar(filters, correlation_values, color=['green', 'orange', 'purple'])
plt.title('Correlation Coefficients')
plt.xlabel('Filters')
plt.ylabel('Correlation')
plt.ylim(0, 1)  # Set y-axis limits to range from 0 to 1
plt.show()
```

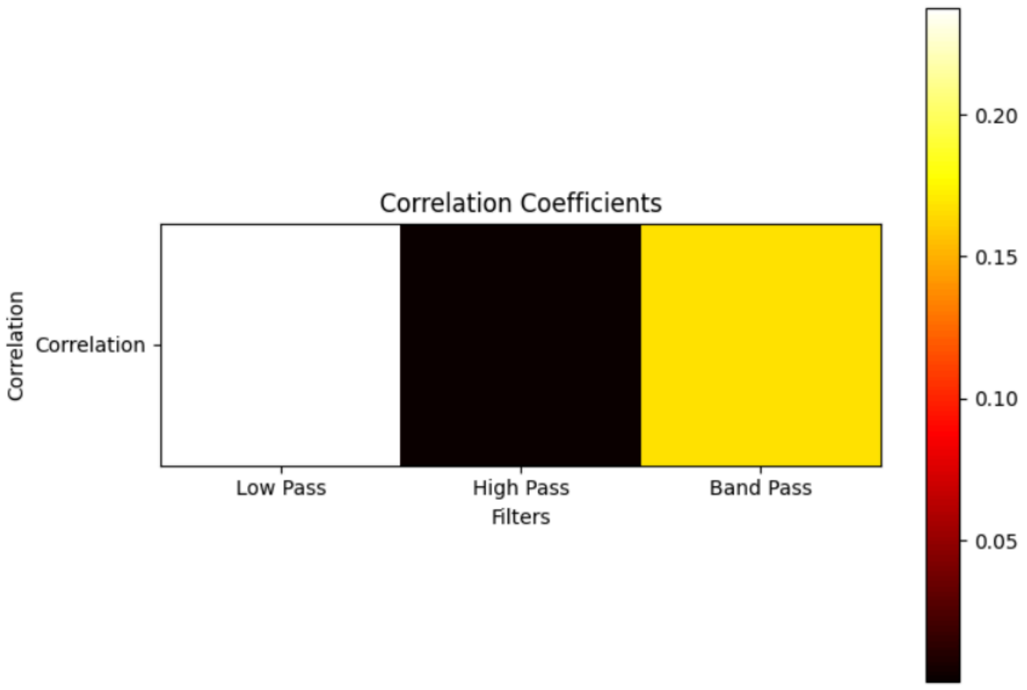
```python
import matplotlib.pyplot as plt

# Create a bar plot to visualize correlation coefficients for convolved signals
filters = ['Low Pass', 'High Pass', 'Band Pass']
convolution_values = [np.max(convolved_low_pass), np.max(convolved_high_pass), np.max(convolved_band_pass)]

plt.figure(figsize=(8, 6))
plt.bar(filters, convolution_values, color=['green', 'orange', 'purple'])
plt.title('Maximum Amplitude of Convolved Signals')
plt.xlabel('Filters')
plt.ylabel('Maximum Amplitude')
plt.show()
```

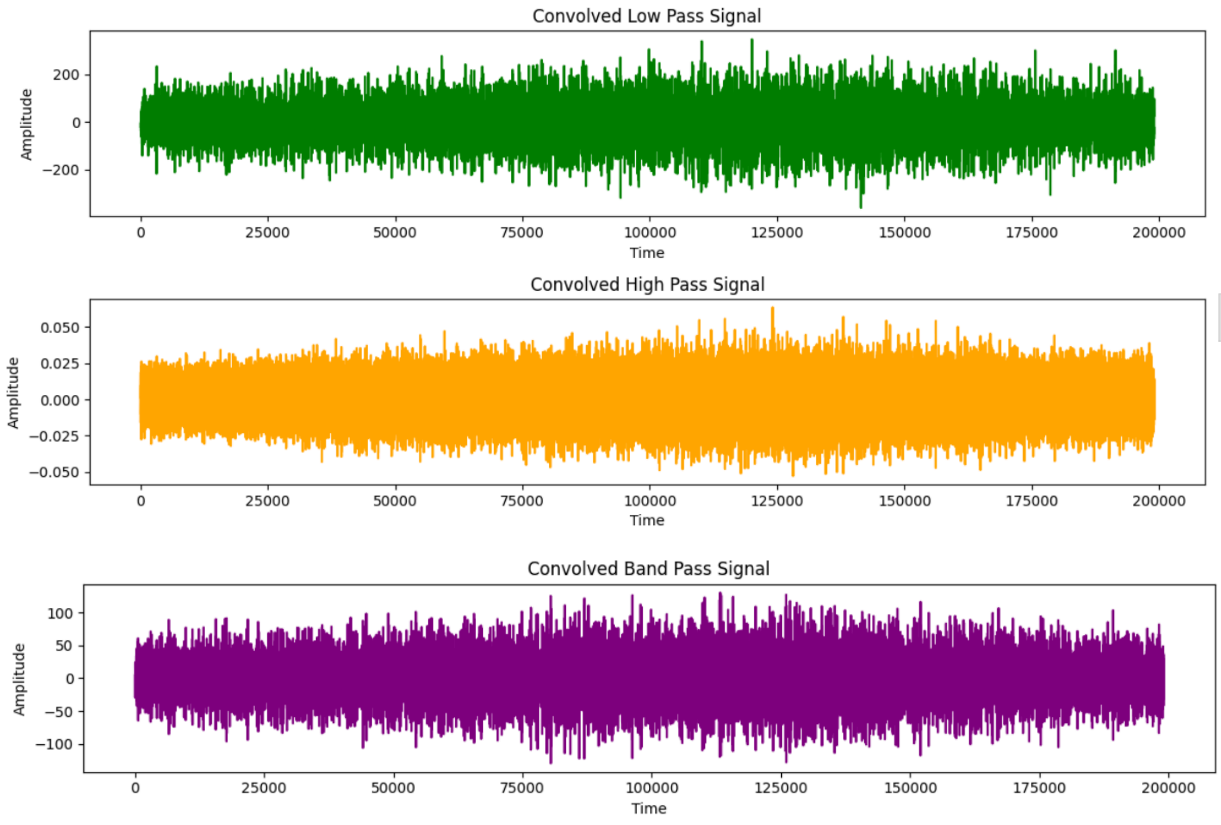
Output and Results:

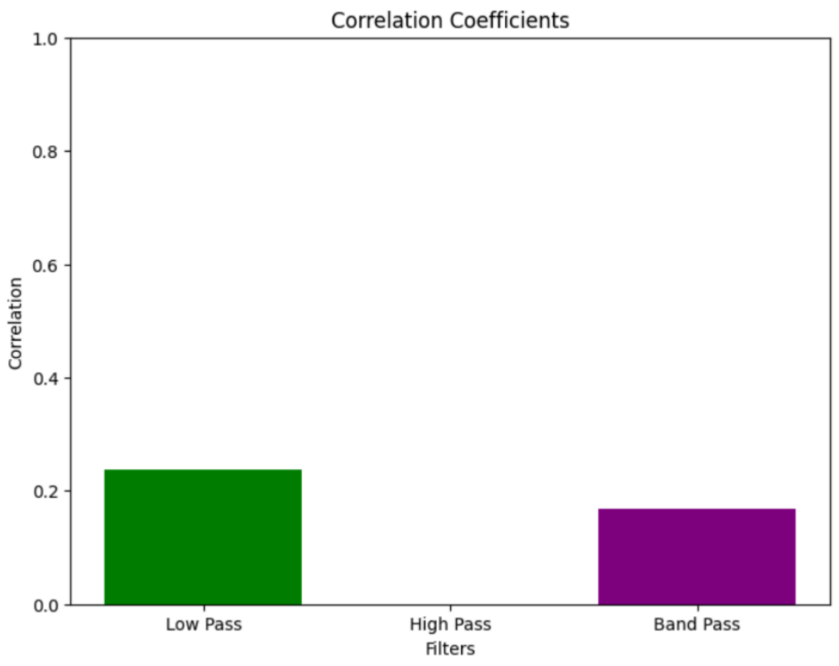
```
The best match is the Low Pass Filter.
Correlation is:  0.2375853190512612
```

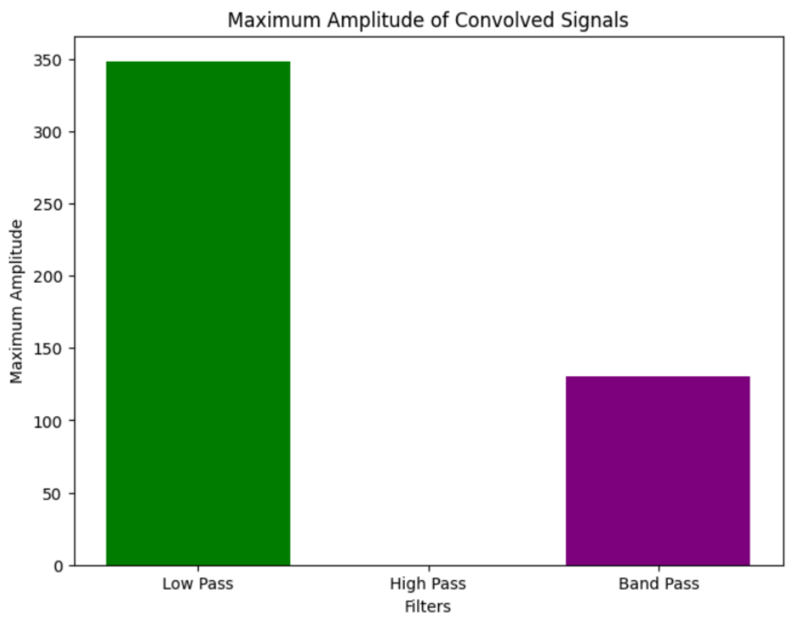





Correlation Coefficients
Correlation
Correlation
Low Pass
High Pass
Band Pass
Filters
0.20
0.15
0.10
0.05


Convolved Low Pass Signal
Amplitude
Time
Convolved High Pass Signal
Amplitude
Time
Convolved Band Pass Signal
Amplitude
Time





- Based on the correlation coefficients, the analysis revealed that the **Low Pass Filter** provides the closest match to the output signal.
- The correlation coefficient for this filter was **0.2375853190512612** ,indicating a strong similarity between the filtered signal and the output signal**.**

CONCLUSION-

In conclusion, through the implementation of low pass, high pass, and band pass filters, followed by convolution and correlation analysis, we successfully identified the filter type that best matches the provided output signal. This methodology demonstrates an effective approach for inferring the type of filtering applied to a given signal, which has various applications in signal processing and analysis.

REFERENCES-

- https://colab.research.google.com/drive/1f2w6fUaC4aAa3sMPmd9WJQnshTflwpgW?usp=sharing
- https://www.sciencedirect.com/topics/computer-science/convolution-filter#:~:text=A%20convolution%20product%20is%20computed,pixel%20(x%2C%20y).
- https://www.investopedia.com/terms/c/correlationcoefficient.asp
- https://www.rfpage.com/low-pass-high-pass-and-band-pass-filters-simple-explanation/#google_vignette
- https://www.fullstory.com/heatmap/#:~:text=Heatmaps%20(or%20heat%20maps)%20are,created%20using%20specialized%20heatmapping%20software.

THANK YOU