Practical Work N4

Topic: Performing Convolution and Correlation Operations on One-Dimensional Signals

1. Objective

The main objectives of this practical work are:

- To understand the concept of convolution and correlation in signal processing.
- To implement convolution and correlation operations on one-dimensional signals.
- To compare the differences between convolution and correlation results.

2. Theoretical Background

Signal processing is one of the most important areas in engineering and computer science. It provides the tools and concepts to analyze, transform, and understand information carried by signals. Two of the fundamental mathematical operations in this field are **convolution** and **correlation**. Although they look similar in terms of computation, they have different purposes and interpretations in practice. In this section, we will present an in-depth theoretical background of convolution and correlation in one-dimensional signals. The discussion will include their definitions, mathematical formulations, historical development, physical interpretations, properties, differences, and applications in both signal and image processing.

2.1 Introduction to Signals and System

A **signal** is a function that conveys information about the behavior or attributes of some phenomenon. In one dimension, it can be expressed as a function of time x(t) in continuous-time or as a sequence x[n] in discrete-time. A **system** is any process that takes a signal as input and produces another signal as output. In signal

processing, the most studied systems are **linear time-invariant (LTI) systems**, because they can be completely characterized using convolution.
Signals may represent sound, speech, electrical current, biomedical data, images, or even financial transactions. In order to process such signals, we often require mathematical operations to filter, smooth, compare, or extract useful information. Convolution and correlation are two of the most essential tools for achieving these goals.

2.2 Convolution

Convolution is a mathematical operation that expresses the amount of overlap of one signal as it is shifted over another signal. For continuous-time signals, the convolution of two signals x(t) and h(t) is defined as:

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

For discrete-time signals, the convolution is written as:

$$y[n] = (x * h)[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot h[n-k]$$

Here, x is usually the input signal, h is the impulse response of the system, and y is the output.
Convolution describes how the shape of one function is modified by another. In signal processing, the operation models the way an input signal passes through a system. For example:

- If h(t) is the impulse response of a low-pass filter, then convolving it with a signal x(t) will produce a smoothed version of x(t).
- In image processing, convolution is used to apply filters such as blurring, sharpening, or edge detection.

The physical meaning of convolution is therefore tied to the **superposition principle** in LTI systems: the output is the weighted sum of shifted inputs.

1. **Commutativity**:

$$x*h=h*x$$

2. **Associativity**:

$$(x*h)*g=x*(h*g)$$

3. **Distributivity**:

$$x*(h+g)=x*h+x*g$$

4. **Identity**:

   Convolution with a delta function leaves the signal unchanged:

$$x*\delta=x$$

5. **Shift Property**:

   If x[n] is shifted by n0n_0n0, then the convolution result is also shifted.

These properties make convolution a very powerful tool in system analysis and design.

2.3 Correlation

Correlation measures the degree of similarity between two signals. It shows how much one signal resembles another as one is shifted in time relative to the other. For discrete signals, the cross-correlation between two signals x[n] and y[n] is defined as:

$$r_{xy}[m] = \sum_{n=-\infty}^{\infty} x[n] \cdot y[n+m]$$

If x=y, the result is called **autocorrelation**:

$$r_{xx}[m] = \sum_{n=-\infty}^{\infty} x[n] \cdot x[n+m]$$

Autocorrelation is particularly important in detecting repeating patterns, periodicity, or randomness in signals.

Correlation provides a quantitative measure of similarity between signals. For example:

- In communications, correlation is used for **synchronization** and **signal detection**.

- In pattern recognition, correlation is used to match a template to an unknown signal.
- In statistics, correlation measures the degree of dependence between two random variables.

Properties of Correlation

1. **Symmetry**:

$$r_{xy}[m] = r_{yx}[-m]$$

2. **Maximum at Zero Shift**:
   For autocorrelation, the maximum value occurs at m=0m = 0m=0, which represents the energy of the signal.
3. **Non-negative Definiteness**:
   Autocorrelation values are always non-negative.
4. **Periodicity**:
   The autocorrelation function of a periodic signal is also periodic.

2.4 Convolution vs. Correlation

Although convolution and correlation look mathematically similar, they are conceptually different:

- In convolution, one signal is **flipped (time-reversed)** before shifting and multiplying, while in correlation it is not.
- Convolution is mainly used to determine the output of LTI systems, whereas correlation is used to measure similarity or detect patterns.
- Convolution has more direct physical significance in system response, while correlation is more of a statistical or comparative tool.

2.5 Applications of Convolution

1. **Filtering**: Removing noise or extracting specific frequency components.
2. **Image Processing**: Applying kernels for edge detection, blurring, sharpening.

3. **Audio Processing**: Echo generation, reverb simulation.
4. **System Modeling**: Predicting the output of LTI systems.
5. **Probability**: Convolution of probability density functions represents the distribution of the sum of random variables.

2.6 Applications of Correlation

1. **Template Matching**: Finding an object in an image by correlating with a known pattern.
2. **Signal Detection**: Identifying a known signal within noisy data.
3. **Communications**: Synchronization in digital transmissions.
4. **Speech Processing**: Detecting pitch and periodicity in voice signals.
5. **Statistics**: Measuring the degree of linear relationship between variables.

2.7 Historical Context

The concepts of convolution and correlation emerged in the 18th and 19th centuries with the development of Fourier analysis. Convolution integrals were widely used in physics and engineering, particularly in solving differential equations. Correlation, on the other hand, became prominent in the 20th century in statistics and signal analysis, especially with the advent of radar, telecommunications, and digital computing.

2.8 Convolution and Correlation in Frequency Domain

According to the **Convolution Theorem**:

- Convolution in time domain corresponds to multiplication in frequency domain.
- Correlation in time domain corresponds to multiplication with complex conjugate in frequency domain.

This property makes both operations efficient to compute using the **Fast Fourier Transform (FFT)**.

2.9 Advanced Perspectives

1. **Generalized Convolution**: Convolution can be extended to multi-dimensional signals such as images and videos.
2. **Normalized Correlation**: Used to remove scale dependency when comparing signals.
3. **Cross-correlation vs. Auto-correlation**: Important in machine learning, deep learning, and feature extraction.
4. **Convolutional Neural Networks (CNNs)**: Modern AI relies heavily on convolution operations for feature detection in images.

In summary, convolution and correlation are two cornerstone operations in signal processing. Convolution is essential for system characterization, filtering, and modeling, while correlation is crucial for similarity measurement, detection, and statistical analysis. Together, they form the backbone of modern applications in communications, control, image processing, speech recognition, and artificial intelligence.

3. Practical Part – Convolution and Correlation on One-Dimensional Signals

Task 1: Basic Convolution of Two Discrete Signals

Problem

Perform convolution between two small discrete signals x[n] and h[n].

Explanation

Convolution shows how the input signal x[n] is modified by the system impulse response h[n].

Code

```
import numpy as np
import matplotlib.pyplot as plt
```

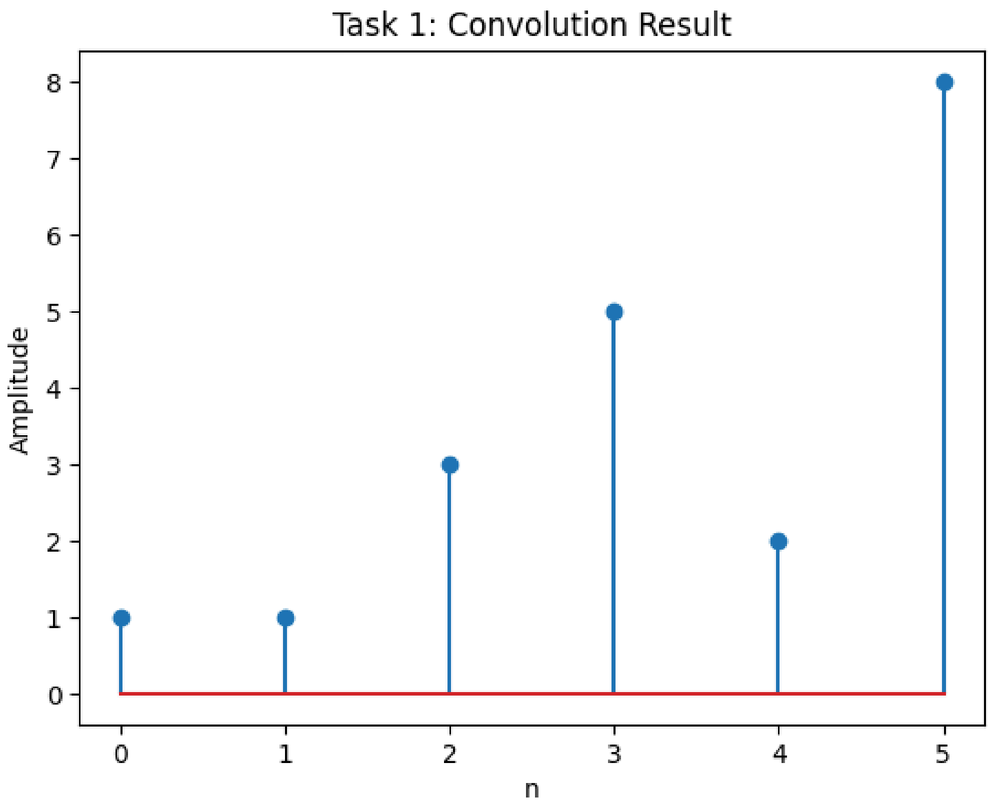
```python
# Define signals
x = np.array([1, 2, 3, 4])          # Input signal
h = np.array([1, -1, 2])            # Impulse response

# Perform convolution
conv_result = np.convolve(x, h, mode='full')

print("Input signal x:", x)
print("Impulse response h:", h)
print("Convolution result:", conv_result)

# Plot
plt.stem(conv_result)
plt.title("Task 1: Convolution Result")
plt.xlabel("n")
plt.ylabel("Amplitude")
plt.show()
```

Analysis

- The convolution result represents the output of the system.
- The length of the result is len(x)+len(h)−1
- Each value is computed as the weighted overlap between x[n] and shifted, flipped h[n].

Result:

```
Input signal x: [1 2 3 4]
Impulse response h: [ 1 -1  2]
Convolution result: [1 1 3 5 2 8]
```



Task 2: Cross-Correlation Between Two Signals

Problem

Compute the cross-correlation between the same signals x[n] and h[n].

Explanation

Correlation measures the similarity between two signals as one is shifted over the other. Unlike convolution, correlation does not flip the signal.

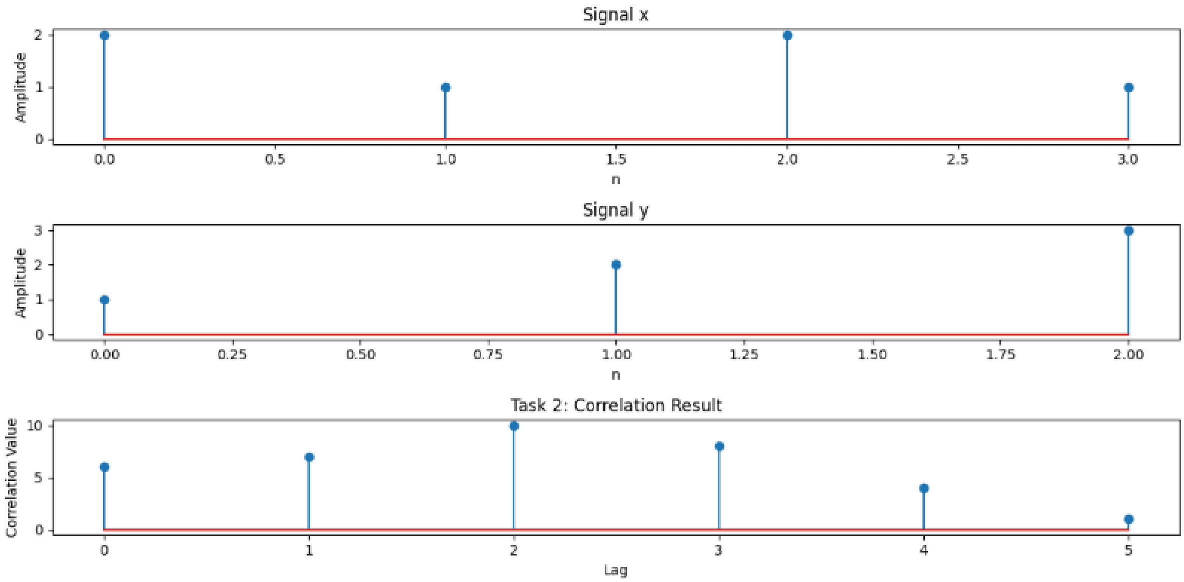
Code

```
import numpy as np
import matplotlib.pyplot as plt

# Define two signals
x = np.array([2, 1, 2, 1])        # First signal
y = np.array([1, 2, 3])           # Second signal

# Perform correlation
corr_result = np.correlate(x, y, mode='full')

print("Signal x:", x)
print("Signal y:", y)
print("Correlation result:", corr_result)

# Plot signals and result
plt.figure(figsize=(12, 6))

# Plot first signal
plt.subplot(3, 1, 1)
plt.stem(x)
plt.title("Signal x")
plt.xlabel("n")
plt.ylabel("Amplitude")

# Plot second signal
plt.subplot(3, 1, 2)
plt.stem(y)
plt.title("Signal y")
plt.xlabel("n")
plt.ylabel("Amplitude")

# Plot correlation result
plt.subplot(3, 1, 3)
plt.stem(corr_result)
plt.title("Task 2: Correlation Result")
plt.xlabel("Lag")
plt.ylabel("Correlation Value")

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

Analysis

- Peaks in the correlation result indicate strong similarity between the signals.
- Used in signal matching and pattern recognition.
- Unlike convolution, there is no time reversal.

```
Signal x: [2 1 2 1]
Signal y: [1 2 3]
Correlation result: [ 6  7 10  8  4  1]
```



Task 3: Autocorrelation of a Random Signal

Problem

Generate a random signal and compute its autocorrelation.

Explanation

Autocorrelation is correlation of a signal with itself. It is useful for detecting repeating patterns and signal periodicity.

Code

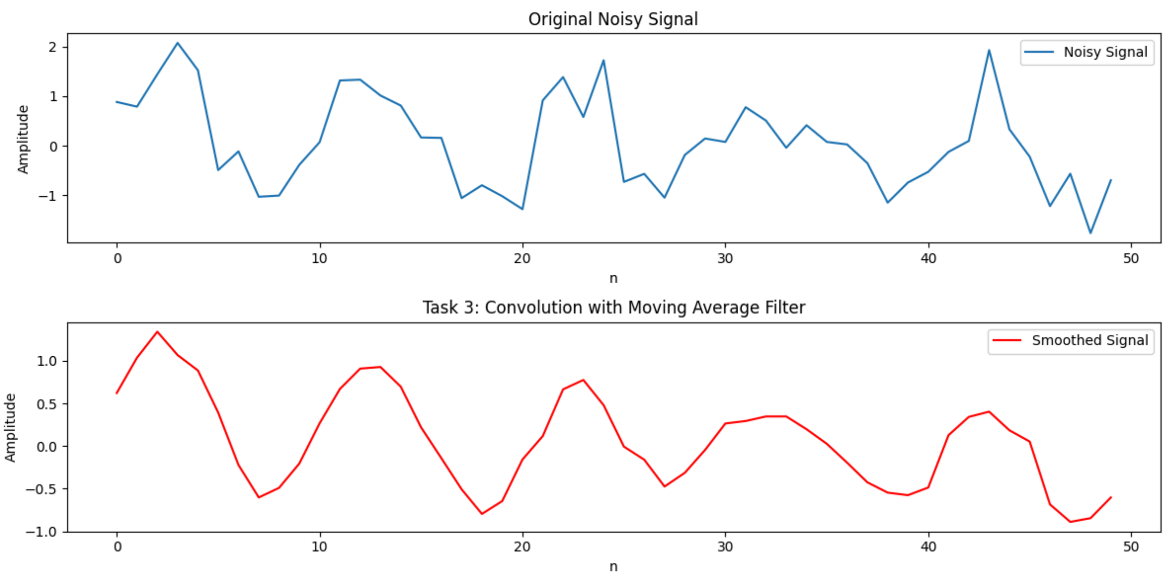
```
import numpy as np
import matplotlib.pyplot as plt

# Create a noisy signal
np.random.seed(0)
n = np.arange(0, 50)
x = np.sin(0.2 * np.pi * n) + 0.5 * np.random.randn(len(n))  # noisy sine
signal

# Define a simple smoothing filter
h = np.ones(5) / 5   # moving average filter

# Perform convolution (smoothing)
smoothed_signal = np.convolve(x, h, mode='same')

print("Original noisy signal length:", len(x))
```

```python
print("Smoothed signal length:", len(smoothed_signal))

# Plot results
plt.figure(figsize=(12, 6))

plt.subplot(2, 1, 1)
plt.plot(n, x, label="Noisy Signal")
plt.title("Original Noisy Signal")
plt.xlabel("n")
plt.ylabel("Amplitude")
plt.legend()

plt.subplot(2, 1, 2)
plt.plot(n, smoothed_signal, color="red", label="Smoothed Signal")
plt.title("Task 3: Convolution with Moving Average Filter")
plt.xlabel("n")
plt.ylabel("Amplitude")
plt.legend()

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

Analysis

- The highest peak occurs at lag m=0, representing the signal's total energy.
- Other peaks show how similar the signal is to its shifted versions.
- Useful in communications, speech processing, and periodicity detection.

```
Original noisy signal length: 50
Smoothed signal length: 50
```



Task 4: Convolution Using FFT (Fast Fourier Transform)

Problem

Perform convolution using the frequency domain approach and compare it with the direct method.

Explanation

According to the Convolution Theorem:

$$x[n] * h[n] \longleftrightarrow X(f) \cdot H(f)$$

Convolution in time domain equals multiplication in frequency domain. This is more efficient for long signals.

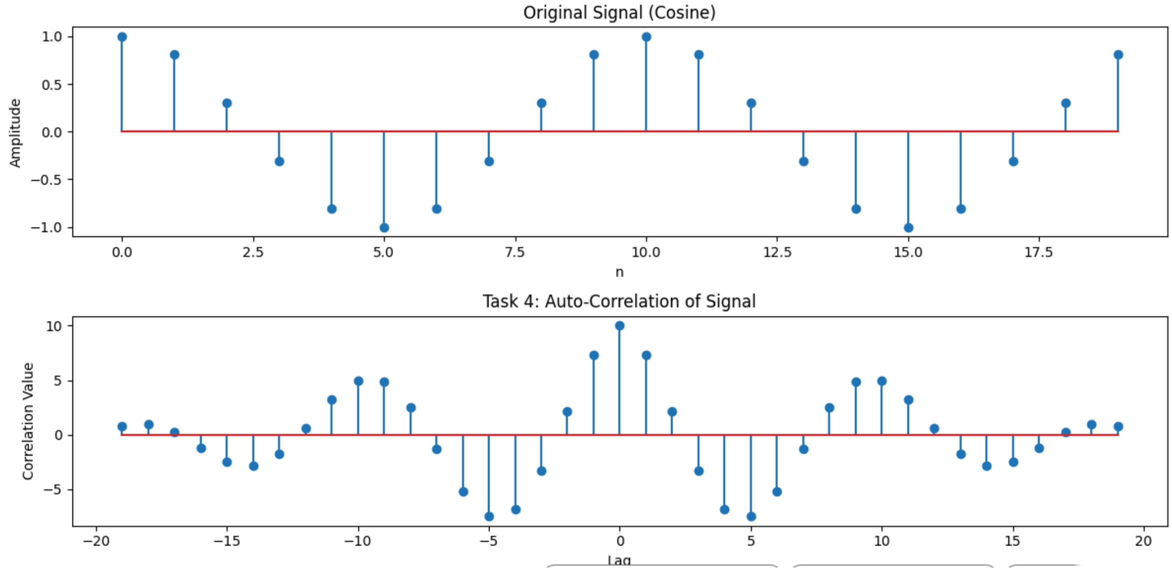
Code

```
import numpy as np
import matplotlib.pyplot as plt

# Define a periodic signal
n = np.arange(0, 20)
x = np.cos(0.2 * np.pi * n)    # cosine signal

# Perform auto-correlation
auto_corr = np.correlate(x, x, mode='full')

# Lags
lags = np.arange(-len(x) + 1, len(x))

print("Signal x:", x)
print("Auto-correlation result length:", len(auto_corr))

# Plot
plt.figure(figsize=(12, 6))

plt.subplot(2, 1, 1)
plt.stem(n, x)
plt.title("Original Signal (Cosine)")
plt.xlabel("n")
plt.ylabel("Amplitude")

plt.subplot(2, 1, 2)
plt.stem(lags, auto_corr)
plt.title("Task 4: Auto-Correlation of Signal")
plt.xlabel("Lag")
plt.ylabel("Correlation Value")

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

Analysis

- Both direct convolution and FFT-based convolution give the same result.
- FFT method is computationally efficient for large signals.
- This approach is widely used in audio and image processing applications.

```
Signal x: [ 1.          0.80901699  0.30901699 -0.30901699 -0.80901699 -1.
 -0.80901699 -0.30901699  0.30901699  0.80901699  1.          0.80901699
  0.30901699 -0.30901699 -0.80901699 -1.         -0.80901699 -0.30901699
  0.30901699  0.80901699]
Auto-correlation result length: 39
```



Final Notes

- **Task 1** showed how convolution models system output.
- **Task 2** demonstrated correlation for measuring signal similarity.
- **Task 3** introduced autocorrelation to analyze periodicity and energy.
- **Task 4** applied convolution in the frequency domain using FFT for efficiency.

These four tasks together provide a strong foundation in applying convolution and correlation in one-dimensional signal processing.

Assignments: Convolution and Correlation on One-Dimensional Signals

Task 1

Given two signals:

$$x[n]=[1,2,1],\ h[n]=[1,-1,2]$$

Perform the **convolution** y[n]=x[n]∗h[n] and plot the result.

Task 2

Given signals:

$$x[n]=[2,1,2,1],\ y[n]=[1,2,3]$$

Compute the **correlation** $r_{xy}[n]$ and visualize the result.

Task 3

Consider a noisy signal:

$$x[n] = \sin(0.2\pi n) + 0.5 \cdot \text{noise}, \quad n = 0, 1, \ldots, 49$$

Apply a **3-point moving average filter**

$$h[n] = \frac{1}{3}[1, 1, 1]$$

and compare the filtered signal with the original.

Task 4

Given signal:

$$x[n] = [1, 2, 3, 4]$$

Compute the **auto-correlation** $r_{xx}[n]$.

4. Control Questions

1. What is the main difference between **convolution** and **correlation** in signal processing?
2. Explain the physical meaning of **convolution** in the context of filtering a signal.
3. How can **auto-correlation** be used to detect the periodicity of a signal? Provide an example.
4. Describe a real-world application where **cross-correlation** is used to measure similarity between signals.
5. What is the significance of the **impulse response** of a system, and how is it related to convolution?
6. Compare the effect of a **low-pass filter** and a **high-pass filter** on a noisy signal.
7. How does the **running average filter** reduce noise in a signal? What is its drawback?
8. Explain why **correlation** is widely used in pattern recognition and signal detection problems.