Experiment 1 : Discrete Convolution

Name	Prathamesh Mane
UID no. & Branch	**2022200078 (B1)**
Experiment No.	1

AIM:	The aim of this experiment is to study mathematical operation such as : • Linear Convolution • Circular Convolution • Linear Convolution using Circular Convolution.
OBJECTIVE:	• To Develop a function to find Linear Convolution and Circular Convolution • To Calculate Linear convolution, Circular convolution, Linear Convolution using Circular Convolution and verify the results using mathematical formulation. • To Conclude on aliasing effect in Circular convolution
INPUT SPECIFICATION:	• Length of first Signal L and signal values. • Length of second Signal M and signal values.
PROBLEM DEFINITION:	• Find Linear Convolution and Circular Convolution of L point sequence x[n] and M point sequence h[n]. • Find Linear Convolution of L point sequence x[n] and M point sequence h[n] using Circular convolution. • Give your conclusion about No of values in Linearly Convolved signal, Aliasing effect in Circular Convolution.
THEORY:	**1. Introduction to Convolution** Convolution is a fundamental mathematical operation used in signal processing, system analysis, and many areas of engineering and physics. It describes how the shape of one function is modified by another function. Convolution can be linear or circular, depending on how the signals are treated during the process.

2. Linear Convolution

Linear convolution is a process where two sequences (or functions) are combined to produce a third sequence that represents the overlap between the two sequences as one slides over the other. Mathematically, the linear convolution of two sequences x[n]x[n]x[n] and h[n]h[n]h[n] is given by:

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

Where:

- x[n] is the input signal.
- h[n] is the impulse response of the system.
- y[n] is the output signal.

Linear convolution is essential in signal processing as it represents the response of a linear time-invariant (LTI) system to an arbitrary input signal.

3. Circular Convolution

Circular convolution is used when the sequences are treated as periodic. It is particularly useful in digital signal processing, especially when dealing with discrete Fourier transforms (DFTs). The circular convolution of two sequences x[n] and h[n] with period N is given by:

$$y[n] = (x \circledast h)[n] = \sum_{k=0}^{N-1} x[k] \cdot h[(n-k) \mod N]$$

Where:

- N is the length of the sequences.
- The modulo operation (n−k) mod N ensures the circular nature of the convolution.
- Circular convolution assumes that the sequences are periodic, so the result may differ from linear convolution unless specific conditions are met.

4. Relationship Between Linear and Circular Convolution

Linear convolution can be computed using circular convolution by padding the sequences with zeros. If two sequences of length N1 and N2 are linearly convolved, the result is a sequence of length N1+N2−1. To calculate this using circular convolution, we can:

- Pad both sequences with zeros to a length of N1+N2−1
- Perform circular convolution on the padded sequences.

5. Mathematical Verification

To verify the results, we compare the outputs of linear convolution, circular convolution, and linear convolution computed using circular convolution:

- **Linear Convolution:** Directly computed using the definition.
- **Circular Convolution:** Computed for the original sequences without zero-padding.
- **Linear Convolution via Circular Convolution:** Computed after zero-padding the sequences and then performing circular convolution.

Case 1 : Liner convolution
(Theoretical solution using formula)

Question : Perform linear convolution of two signal A={12,13,10,11} and B={9,7,14,2}

Solution:

Step 1: Identify the Length of the Resultant Sequence y

The length of the linear convolution result y[n] is given by:

$N=L+M-1$
where L and M are the lengths of sequences A and B, respectively.

For A={12,13,10,11} L=4.

For B={9,7,14,2} M=4

So, the length of the resultant sequence is: N=4+4-1 = 7

Step 2: Lets Write the Sequences in Tabular Form

Align zero point of the sequence A and b together then flip the signal B and do multiply and add operation while shifting the sequence to the right by one unit.

Step 3: Perform the Convolution

We calculate each term of the convolution by multiplying and summing the aligned elements.

1. y[0]

   $y[0] = 12 \times 9$
   $= 108$

2. y[1]

   $y[1] = (12 \times 7)+(13 \times 9)$
   $= 84+117$
   $= 201$

3. y[2]

   $y[2] = (12 \times 14)+(13 \times 7)+(10 \times 9)$
   $= 168+91+90$
   $= 349$

4. y[3]

   $y[3] = (12 \times 2)+(13 \times 14)+(10 \times 7)+(11 \times 9)$
   $= 24+182+70+99$
   $= 375$

5. y[4]

   $y[4] = (13 \times 2)+(10 \times 14)+(11 \times 7)$
   $= 26+140+77$
   $= 243$

6. y[5]

   $y[5] = (10 \times 2)+(11 \times 14)$
   $= 20+154$
   $= 174$

7. y[6]

   y[6] = 11×2
        = 22

Step 4: Final Result

The linear convolution of A={12,13,10,11} and B={9,7,14,2}

y[n]={108,201,349,375,243,174,22}

Case 1 : Linear convolution (Developing algorithm using programming language)	
PROBLEM STATEMENT:	Write a program in any programming language to perform the linear convolution of two signals with length L and M respectively.

PROGRAM:	#include <stdio.h>
	
	// Function to perform linear convolution of two signals
	void LC(float inputSignal[], int lengthX, float impulseResponse[], int lengthH, float outputSignal[]) {
	
	  int i, j;
	
	  // Iterate through each element of the output signal
	  for (i = 0; i < lengthX + lengthH - 1; i++) {
	  outputSignal[i] = 0; // Initialize each element to zero
	
	  for (j = 0; j < lengthX; j++) {
	  if (i - j >= 0 && i - j < lengthH) {
	  outputSignal[i] += inputSignal[j] * impulseResponse[i - j];
	  }
	  }
	  }
	}
	
	int main() {
	
	  int index, lengthX, lengthH, outputLength;
	  float signal[10], response[10], result[20]; // Arrays for input signals and output result

```
// Initialize all arrays to zero
for (index = 0; index < 10; index++) {
   signal[index] = response[index] = 0.0;
}

for (index = 0; index < 20; index++) {
   result[index] = 0.0;
}

// Get the length and values for input signal x[n]
printf("\nEnter the length of the input signal x[n] (L): ");
scanf("%d", &lengthX);

printf("Enter the values for x[n]:\n");
for (index = 0; index < lengthX; index++) {
   scanf("%f", &signal[index]);
}

// Get the length and values for the impulse response h[n]
printf("\nEnter the length of the impulse response h[n] (M): ");
scanf("%d", &lengthH);

printf("Enter the values for h[n]:\n");
for (index = 0; index < lengthH; index++) {
   scanf("%f", &response[index]);
}

// Calculate the length of the output signal y[n]
outputLength = lengthX + lengthH - 1;

// Perform the linear convolution
LC(signal, lengthX, response, lengthH, result);

// Display the input signal x[n]
printf("\n\nx[n] = ");
for (index = 0; index < lengthX; index++) {
   printf("  %1.2f  ", signal[index]);
}

// Display the impulse response h[n]
printf("\n\nh[n] = ");
for (index = 0; index < lengthH; index++) {
   printf("  %4.2f  ", response[index]);
}

// Display the output signal y[n]
```

```
    printf("\n\ny[n] = ");
    for (index = 0; index < outputLength; index++) {
        printf("  %4.2f  ", result[index]);
    }

    printf("\n\n");

    return 0;
}
```

RESULT:

```
/tmp/fVvxpSxecg.o

Enter the length of the input signal x[n] (L): 4
Enter the values for x[n]:
12 13 10 11

Enter the length of the impulse response h[n] (M): 4
Enter the values for h[n]:
9 7 14 2


x[n] =   12.00    13.00    10.00    11.00

h[n] =   9.00    7.00    14.00    2.00

y[n] =   108.00    201.00    349.00    375.00    243.00    174.00    22.00
```

```
/tmp/kkjsKrW9yJ.o

Enter the length of the input signal x[n] (L): 4
Enter the values for x[n]:
23 12 13 17

Enter the length of the impulse response h[n] (M): 3
Enter the values for h[n]:
9 8 7


x[n] =   23.00    12.00    13.00    17.00

h[n] =   9.00    8.00    7.00

y[n] =   207.00    292.00    374.00    341.00    227.00    119.00
```

Case 2 : Circular convolution (Theoretical solution using formula)

Question : To perform circular convolution between two sequences **A = [4, 6, 9, 10]**
B = [3, 2, 1, 11, 7]

Solution :

To perform circular convolution between two sequences, both sequences must be of the same length. Since sequence `A` has 4 elements and sequence `B` has 5 elements, we'll first need to pad the shorter sequence `A` with zeros to match the length of `B`.

Step 1: Pad the shorter sequence

- **A = [4, 6, 9, 10]** becomes **A' = [4, 6, 9, 10, 0]**
- **B = [3, 2, 1, 11, 7]**

Now, both sequences are of length 5.

Step 2: Perform circular convolution

Let's calculate the circular convolution step-by-step:

- y[0] = A[0] * B[0] + A[1] * B[4] + A[2] * B[3] + A[3] * B[2] + A[4] * B[1]
  y[0] = (4 * 3) + (6 * 7) + (9 * 11) + (10 * 1) + (0 * 2)
  y[0] = 12 + 42 + 99 + 10 + 0 = 163

- y[1] = A[0] * B[1] + A[1] * B[0] + A[2] * B[4] + A[3] * B[3] + A[4] * B[2]
  y[1] = (4 * 2) + (6 * 3) + (9 * 7) + (10 * 11) + (0 * 1)
  y[1] = 8 + 18 + 63 + 110 + 0 = 199

- y[2] = A[0] * B[2] + A[1] * B[1] + A[2] * B[0] + A[3] * B[4] + A[4] * B[3]
  y[2] = (4 * 1) + (6 * 2) + (9 * 3) + (10 * 7) + (0 * 11)
  y[2] = 4 + 12 + 27 + 70 + 0 = 113

- y[3] = A[0] * B[3] + A[1] * B[2] + A[2] * B[1] + A[3] * B[0] + A[4] * B[4]
  y[3] = (4 * 11) + (6 * 1) + (9 * 2) + (10 * 3) + (0 * 7)
  y[3] = 44 + 6 + 18 + 30 + 0 = 98

- y[4] = A[0] * B[4] + A[1] * B[3] + A[2] * B[2] + A[3] * B[1] + A[4] * B[0]
  y[4] = (4 * 7) + (6 * 11) + (9 * 1) + (10 * 2) + (0 * 3)
  y[4] = 28 + 66 + 9 + 20 + 0 = 123

Step 3 : Final Result:

The circular convolution result is: **y = [163, 199, 113, 98, 123]**

Case 2 : Circular convolution (Developing algorithm using programming language)	
PROBLEM STATEMENT:	Write a program in any programming language to perform the circular convolution of two signals with length L and M respectively.

PROGRAM:	#include <stdio.h>
	
	void circularConvolution(float signalA[], float signalB[], int N, float result[]) {
	  int i, j;
	  float sum;
	
	  // Perform circular convolution
	  for (i = 0; i < N; i++) {
	  sum = 0.0;
	  for (j = 0; j < N; j++) {
	  int index = (i - j + N) % N; // Handle circular indexing
	  sum = sum + signalA[j] * signalB[index];
	  }
	  result[i] = sum;
	  }
	}
	
	int main() {
	  int lengthA, lengthB, N;
	  float signalA[20], signalB[20], output[20];
	
	  // Initialize arrays to zero
	  for (int i = 0; i < 20; i++) {
	  signalA[i] = signalB[i] = output[i] = 0.0;
	  }
	
	  // Get the length and values for the first signal
	  printf("Enter the length of the first signal A: ");
	  scanf("%d", &lengthA);
	  printf("Enter the values of signal A:\n");
	  for (int i = 0; i < lengthA; i++) {
	  scanf("%f", &signalA[i]);
	  }

```
    // Get the length and values for the second signal
    printf("\n\nEnter the length of the second signal B: ");
    scanf("%d", &lengthB);
    printf("Enter the values of signal B:\n");
    for (int i = 0; i < lengthB; i++) {
        scanf("%f", &signalB[i]);
    }

    // Determine the maximum length for circular convolution
    //use tertitary operator
    N = (lengthA > lengthB) ? lengthA : lengthB;

    // Perform circular convolution
    circularConvolution(signalA, signalB, N, output);

    // Display the input signals and the output signal
    printf("\nSignal A = ");
    for (int i = 0; i < N; i++) {
        printf("%4.2f    ", signalA[i]);
    }

    printf("\nSignal B = ");
    for (int i = 0; i < N; i++) {
        printf("%4.2f    ", signalB[i]);
    }

    printf("\n\nCircular Convolution Result y[n] = ");
    for (int i = 0; i < N; i++) {
        printf("%4.2f    ", output[i]);
    }
    printf("\n");

    return 0;
}
```

RESULT:

```
/tmp/irej4L9aLa.o
Enter the length of the first signal A: 4
Enter the values of signal A:
4 6 9 10


Enter the length of the second signal B: 5
Enter the values of signal B:
3 2 1 11 7

Signal A = 4.00    6.00    9.00    10.00    0.00
Signal B = 3.00    2.00    1.00    11.00    7.00

Circular Convolution Result y[n] = 163.00   199.00   113.00   98.00   123.00
```

```
/tmp/W7tDG4qXwr.o
Enter the length of the first signal A: 4
Enter the values of signal A:
7 8 9 10


Enter the length of the second signal B: 4
Enter the values of signal B:
7 8 9 10

Signal A = 7.00    8.00    9.00    10.00
Signal B = 7.00    8.00    9.00    10.00

Circular Convolution Result y[n] = 290.00   292.00   290.00   284.00
```

Case 3 : Linear Convolution using Circular Convolution (Theoretical solution using formula)

Step 1: Zero Padding

Since the circular convolution assumes that the sequences are of the same length, we need to pad the sequences with zeros to make their lengths equal to N+M−1, where N and M are the lengths of the original sequences A and B

Given:

- Signal A: A=[6,9,10,13]
- Signal B: B=[12,8,7,14]

The length of the circular convolution will be N+M−1=4+4−1=7.

Pad both sequences with zeros to make their lengths 7:

- A=[6,9,10,13,0,0,0]
- B=[12,8,7,14,0,0,0]

Step 2: Perform Circular Convolution

Circular convolution is performed by rotating the sequence B and taking the dot product with A for each rotation.Circular convolution y[n] will also have a length of 7.

Compute Each Element:

- $y[0]=(6\times12)+(9\times0)+(10\times0)+(13\times0)+(0\times0)+(0\times0)+(0\times0)$
  $=72$

- $y[1]=(6\times8)+(9\times12)+(10\times0)+(13\times0)+(0\times0)+(0\times0)+(0\times0)$
  $=48+108$
  $=156$

- $y[2]=(6\times7)+(9\times8)+(10\times12)+(13\times0)+(0\times0)+(0\times0)+(0\times0)$
  $=42+72+120$
  $=234$

- $y[3]=(6\times14)+(9\times7)+(10\times8)+(13\times12)+(0\times0)+(0\times0)+(0\times0)$
  $=84+63+80+156$
  $=383$

- $y[4]=(6\times0)+(9\times14)+(10\times7)+(13\times8)+(0\times12)+(0\times0)+(0\times0)$
  $=126+70+104$
  $=300$

- $y[5]=(6\times0)+(9\times0)+(10\times14)+(13\times7)+(0\times8)+(0\times12)+(0\times0)$
  $=140+91$
  $=231$

- $y[6]=(6\times0)+(9\times0)+(10\times0)+(13\times14)+(0\times7)+(0\times8)+(0\times12)$
  $=182$

Final Result:

The linear convolution result using circular convolution method is:

y[n]=[72,156,234,383,300,231,182]

Case 3 : Linear Convolution using Circular Convolution (Developing algorithm using programming language)	
PROBLEM STATEMENT:	Write a program in any programming language to perform the linear convolution using circular convolution of two signals with length L and M respectively.

PROGRAM:	#include <stdio.h>
	
	// Function to perform circular convolution
	void circularConvolution(float signalA[], float signalB[], int N, float result[]) {
	  int i, j;
	  float sum;
	
	  // Perform circular convolution
	  for (i = 0; i < N; i++) {
	  sum = 0.0;
	  for (j = 0; j < N; j++) {
	  int index = (i - j + N) % N; // Handle circular indexing
	  sum = sum + signalA[j] * signalB[index];
	  }
	  result[i] = sum;
	  }
	}

```
// Function to calculate linear convolution using circular convolution
void linearConvolutionViaCircular(float signalA[], float signalB[], int
lengthA, int lengthB, float result[]) {
    int N = lengthA + lengthB - 1;
    float paddedA[20] = {0}, paddedB[20] = {0}, circResult[20] = {0};

    // Zero-padding the signals
    for (int i = 0; i < lengthA; i++) {
        paddedA[i] = signalA[i];
    }
    for (int i = 0; i < lengthB; i++) {
        paddedB[i] = signalB[i];
    }

    // Perform circular convolution on padded signals
    circularConvolution(paddedA, paddedB, N, circResult);

    // Extract the linear convolution result
    for (int i = 0; i < N; i++) {
        result[i] = circResult[i];
    }
}

int main() {
    int lengthA, lengthB, N;
    float signalA[20] = {0}, signalB[20] = {0}, output[20] = {0};

    // Get the length and values for the first signal
    printf("Enter the length of the first signal A: ");
    scanf("%d", &lengthA);
    printf("Enter the values of signal A:\n");
    for (int i = 0; i < lengthA; i++) {
        scanf("%f", &signalA[i]);
    }

    // Get the length and values for the second signal
    printf("\nEnter the length of the second signal B: ");
    scanf("%d", &lengthB);
    printf("Enter the values of signal B:\n");
    for (int i = 0; i < lengthB; i++) {
```

```
        scanf("%f", &signalB[i]);
    }

    // Calculate the length for linear convolution
    N = lengthA + lengthB - 1;

    // Perform linear convolution using circular convolution
    linearConvolutionViaCircular(signalA, signalB, lengthA, lengthB,
output);

    // Display the input signals and the output signal
    printf("\nSignal A = ");
    for (int i = 0; i < lengthA; i++) {
        printf("%4.2f    ", signalA[i]);
    }

    printf("\nSignal B = ");
    for (int i = 0; i < lengthB; i++) {
        printf("%4.2f    ", signalB[i]);
    }

    printf("\n\nLinear Convolution Result y[n] = \n");
    for (int i = 0; i < N; i++) {
        printf("%4.2f    ", output[i]);
    }
    printf("\n");

    return 0;
}
```

RESULT:

```
/tmp/nbUY1kwtCR.o
Enter the length of the first signal A: 4
Enter the values of signal A:
6 9 10 13

Enter the length of the second signal B: 4
Enter the values of signal B:
12 8 7 14

Signal A = 6.00    9.00    10.00    13.00
Signal B = 12.00    8.00    7.00    14.00

Linear Convolution Result y[n] =
72.00   156.00   234.00   383.00   300.00   231.00   182.00
```

```
Enter the length of the first signal A: 4
Enter the values of signal A:
1 2 3 4

Enter the length of the second signal B: 3
Enter the values of signal B:
5 6 7

Signal A = 1.00    2.00    3.00    4.00
Signal B = 5.00    6.00    7.00

Linear Convolution Result y[n] =
5.00   16.00   34.00   52.00   45.00   28.00
```

CONCLUSION:	• Function Accuracy: The developed functions for Linear and Circular Convolution produced accurate results, which matched with mathematical formulations. • Linear Convolution Length: The output length $N=L+M-1$ was correctly calculated, matching theoretical expectations. • Causality Preservation: The output of Linear Convolution remains causal if both input signals are causal. • Circular Convolution Calculation: Using N=Maximum(L,M) accurately computed Circular Convolution, reflecting the periodic nature of the signals. • Linear Convolution via Circular Convolution: Selecting $N \geq L+M-1$ for Circular Convolution correctly produced the Linear Convolution results. • Aliasing in Circular Convolution: Aliasing effects were observed in Circular Convolution, highlighting the need to manage signal length and periodicity to minimize these effects.