**Experiment No:** 03  
**Experiment Name:** Study & Analysis of Convolution and Correlation of Discrete-Time Signals.

**Theory:**In **Digital Signal Processing (DSP),** convolution and correlation are fundamental operations used for analyzing **linear time-invariant (LTI) systems**, filtering, and measuring signal similarity.

***Linear-Convolution***  
The output of an LTI system with input x[n] and impulse response h[n] is:

For sequences of length M and N, the result has length M+N−1.

***Circular-Convolution***  
Assumes signals are periodic with period N:

Steps:

1. Zero-pad *x*[*n*] and *h*[*n*] to length *N*.
2. Compute DFT of both signals.
3. Multiply the DFTs element-wise.
4. Compute the inverse DFT (IDFT) to obtain the circular convolution result

***Cross-Correlation***Measures similarity between two signals as one is shifted:

***Auto-Correlation***Cross-correlation of a signal with itself:

It is useful for identifying periodicity, patterns, and noise properties.

**Required Software:**

* MATLAB

**Code and Output:**Task 1:

|  |  |
| --- | --- |
| % Given sequences  x\_n = [2, 1, -1, 0, 2, 9, 3, 1];  h\_n = [0, 0, 2, 1, 1, 3];  % Linear Convolution  y\_linear = conv(x\_n, h\_n);  % Circular Convolution  % Length for circular convolution (same as linear convolution length)  N = length(x\_n) + length(h\_n) - 1;  % Zero-pad both sequences to length N  x\_pad = [x\_n, zeros(1, N - length(x\_n))];  h\_pad = [h\_n, zeros(1, N - length(h\_n))];  % Circular convolution using FFT method  y\_circular = ifft(fft(x\_pad) .\* fft(h\_pad));  % Cross-Correlation (x[n] and y[n])  [cross\_xy, lags\_xy] = xcorr(x\_n, y\_linear); % cross correlation with lag indices  % Autocorrelation of x[n]  [auto\_x, lags\_x] = xcorr(x\_n); % auto correlation with lag indices  % Display Results  disp('Linear Convolution Result:');  disp(y\_linear);  disp('Circular Convolution Result (Length N):');  disp(real(y\_circular));  disp('Cross-Correlation between x[n] and y[n]:');  disp(cross\_xy);  disp('Autocorrelation of x[n]:');  disp(auto\_x);  % Plot Results  n\_x = 0:length(x\_n)-1; % Time index for x(n)  n\_h = 0:length(h\_n)-1; % Time index for h(n)  n\_y = 0:length(y\_linear)-1; % Time index for convolution results | figure;  subplot(6,1,1);  stem(n\_x, x\_n, 'filled');  title('x(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,2);  stem(n\_h, h\_n, 'filled');  title('h(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,3);  stem(n\_y, y\_linear, 'filled');  title('Linear Convolution: y(n) = x(n) \* h(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,4);  stem(0:N-1, real(y\_circular), 'filled');  title(sprintf('Circular Convolution (N = %d)', N));  xlabel('n'); ylabel('Amplitude');  subplot(6,1,5);  stem(lags\_xy, cross\_xy, 'filled'); % Cross correlation  title('Cross-Correlation: R\_{xy}[k]');  xlabel('Lag k'); ylabel('Amplitude');  subplot(6,1,6);  stem(lags\_x, auto\_x, 'filled'); % Auto correlation  title('Autocorrelation: R\_{xx}[k]');  xlabel('Lag k'); ylabel('Amplitude'); |

**Output:**



Figure 1: Output of Task 1

Task 2:

|  |  |
| --- | --- |
| % Define n range  n = 0:20; % 10 samples  % Define sequences  x\_n = sin(n);  h\_n = cos(n);  % Linear Convolution  y\_linear = conv(x\_n, h\_n);  % Circular Convolution  % Length for circular convolution (same as linear convolution length)  N = length(x\_n) + length(h\_n) - 1;  % Zero-pad both sequences to length N  x\_pad = [x\_n, zeros(1, N - length(x\_n))];  h\_pad = [h\_n, zeros(1, N - length(h\_n))];  % Circular convolution using FFT method  y\_circular = ifft(fft(x\_pad) .\* fft(h\_pad));  % Cross-Correlation (x[n] and y[n])  [cross\_xy, lags\_xy] = xcorr(x\_n, y\_linear);  % Autocorrelation of x[n]  [auto\_x, lags\_x] = xcorr(x\_n);  figure;  subplot(6,1,1);  stem(n\_x, x\_n, 'filled');  title('x(n) = sin(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,2);  stem(n\_h, h\_n, 'filled');  title('h(n) = cos(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,3);  stem(n\_y, y\_linear, 'filled');  title('Linear Convolution: y(n) = x(n) \* h(n)');  xlabel('n'); ylabel('Amplitude'); | % Display Results  disp('x(n) = sin(n):');  disp(x\_n);  disp('h(n) = cos(n):');  disp(h\_n);  disp('Linear Convolution Result:');  disp(y\_linear);  disp('Circular Convolution Result (Length N):');  disp(real(y\_circular));  disp('Cross-Correlation between x[n] and y[n]:');  disp(cross\_xy);  disp('Autocorrelation of x[n]:');  disp(auto\_x);  % Plot Results  n\_x = 0:length(x\_n)-1; % Time index for x(n)  n\_h = 0:length(h\_n)-1; % Time index for h(n)  n\_y = 0:length(y\_linear)-1; % Time index for convolution results  subplot(6,1,4);  stem(0:N-1, real(y\_circular), 'filled');  title(sprintf('Circular Convolution (N = %d)', N));  xlabel('n'); ylabel('Amplitude');  subplot(6,1,5);  stem(lags\_xy, cross\_xy, 'filled');  title('Cross-Correlation: R\_{xy}[k]');  xlabel('Lag k'); ylabel('Amplitude');  subplot(6,1,6);  stem(lags\_x, auto\_x, 'filled');  title('Autocorrelation: R\_{xx}[k]');  xlabel('Lag k'); ylabel('Amplitude'); |

**Output:**



**Figure 2:** Output of Task 2

Task 3:

|  |  |
| --- | --- |
| % Define n range  n = 0:20; % 20 samples  % Define sequences  x\_n = -sin(n);  h\_n = -cos(-n); % same as -cos(n) because cosine is even  % Linear Convolution  y\_linear = conv(x\_n, h\_n);  % Circular Convolution  % Length for circular convolution (same as linear convolution length)  N = length(x\_n) + length(h\_n) - 1;  % Zero-pad both sequences to length N  x\_pad = [x\_n, zeros(1, N - length(x\_n))];  h\_pad = [h\_n, zeros(1, N - length(h\_n))];  % Circular convolution using FFT method  y\_circular = ifft(fft(x\_pad) .\* fft(h\_pad));  % Cross-Correlation (x[n] and y[n])  [cross\_xy, lags\_xy] = xcorr(x\_n, y\_linear);  % Autocorrelation of x[n]  [auto\_x, lags\_x] = xcorr(x\_n);  % Display Results  disp('x(n) = -sin(n):');  disp(x\_n);  disp('h(n) = -cos(-n):');  disp(h\_n);  disp('Linear Convolution Result:');  disp(y\_linear);  disp('Circular Convolution Result (Length N):');  disp(real(y\_circular));  disp('Cross-Correlation between x[n] and y[n]:');  disp(cross\_xy);  disp('Autocorrelation of x[n]:');  disp(auto\_x);  % Plot Results  n\_x = 0:length(x\_n)-1; % Time index for x(n)  n\_h = 0:length(h\_n)-1; % Time index for h(n)  n\_y = 0:length(y\_linear)-1; % Time index for convolution results | figure;  subplot(6,1,1);  stem(n\_x, x\_n, 'filled');  title('x(n) = -sin(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,2);  stem(n\_h, h\_n, 'filled');  title('h(n) = -cos(-n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,3);  stem(n\_y, y\_linear, 'filled');  title('Linear Convolution: y(n) = x(n) \* h(n)');  xlabel('n'); ylabel('Amplitude');  subplot(6,1,4);  stem(0:N-1, real(y\_circular), 'filled');  title(sprintf('Circular Convolution (N = %d)', N));  xlabel('n'); ylabel('Amplitude');  subplot(6,1,5);  stem(lags\_xy, cross\_xy, 'filled');  title('Cross-Correlation: R\_{xy}[k]');  xlabel('Lag k'); ylabel('Amplitude');  subplot(6,1,6);  stem(lags\_x, auto\_x, 'filled');  title('Autocorrelation: R\_{xx}[k]');  xlabel('Lag k'); ylabel('Amplitude'); |

**Output:**

**Figure 3:** Output of Task 3

**Discussion:  
Linear vs. Circular Convolution:**  
Circular convolution becomes identical to linear convolution when zero-padding is applied with N = Lx + Lh − 1, which avoids aliasing [2]. Arbitrary sequences (Task 1) produced impulse-like results, while sinusoids (Tasks 2/3) gave oscillatory outputs. Circular convolution is computationally efficient via FFT for large N, but padding is required for accuracy.

**Convolution vs. Cross-Correlation:**  
Convolution blends signals to form a longer sequence, while cross-correlation measures similarity without time reversal. In Task 1, convolution gave higher peaks (31) compared to correlation (27), reflecting accumulation versus overlap. For sinusoidal inputs, both produced oscillations, but correlation revealed phase shifts (e.g., π/2 lag between sine and cosine).

**Cross-Correlation vs. Auto-Correlation:**  
Auto-correlation is always symmetric and non-negative at k=0, representing signal energy, while cross-correlation can be asymmetric and negative (anti-similarity). In Tasks 2/3, auto-correlation showed cosine-like decay patterns, while cross-correlation highlighted the quadrature (90 degree) relationship between sine and cosine. In Task 1, auto-correlation peaked at zero (value 101), equal to total signal energy.

**Conclusion:**

The experiment successfully demonstrated convolution and correlation operations on discrete-time signals. Results confirmed theoretical expectations: linear and circular convolutions matched with zero-padding, while correlation methods provided insights into similarity, periodicity, and energy properties. These operations are fundamental tools in DSP, with applications in filtering, detection, and pattern recognition.

# References

|  |  |
| --- | --- |
| [1] | J. G. Proakis and D. G. Manolakis, Digital Signal Processing: Principles, Algorithms, and Applications, Upper Saddle River, NJ, USA: Pearson, 2007. |
| [2] | "MathWorks," [Online]. Available: https://www.mathworks.com/help/signal/. [Accessed 29 August 2025]. |