**RollNo: 5117060**

**EXPERIMENT 02**

**Aim:** To implement Discrete Correlation.

**Theory:**

**The Meaning of Correlation**

In general, correlation describes the mutual relationship which exists between two or more things. The same definition holds good even in the case of signals. That is, correlation between signals indicates the measure up to which the given signal resembles another signal.

In other words, if we want to know how much similarity exists between the signals 1 and 2, then we need to find out the correlation of Signal 1 with respect to Signal 2 or vice versa.

**Types of Correlation**

Depending on whether the signals considered for correlation are same or different, we have two kinds of correlation: autocorrelation and cross-correlation.

**Autocorrelation**

This is a type of correlation in which the given signal is correlated with itself, usually the time shifted version of itself. Mathematical expression for the autocorrelation of continuous time signal *x* (*t*) is given by

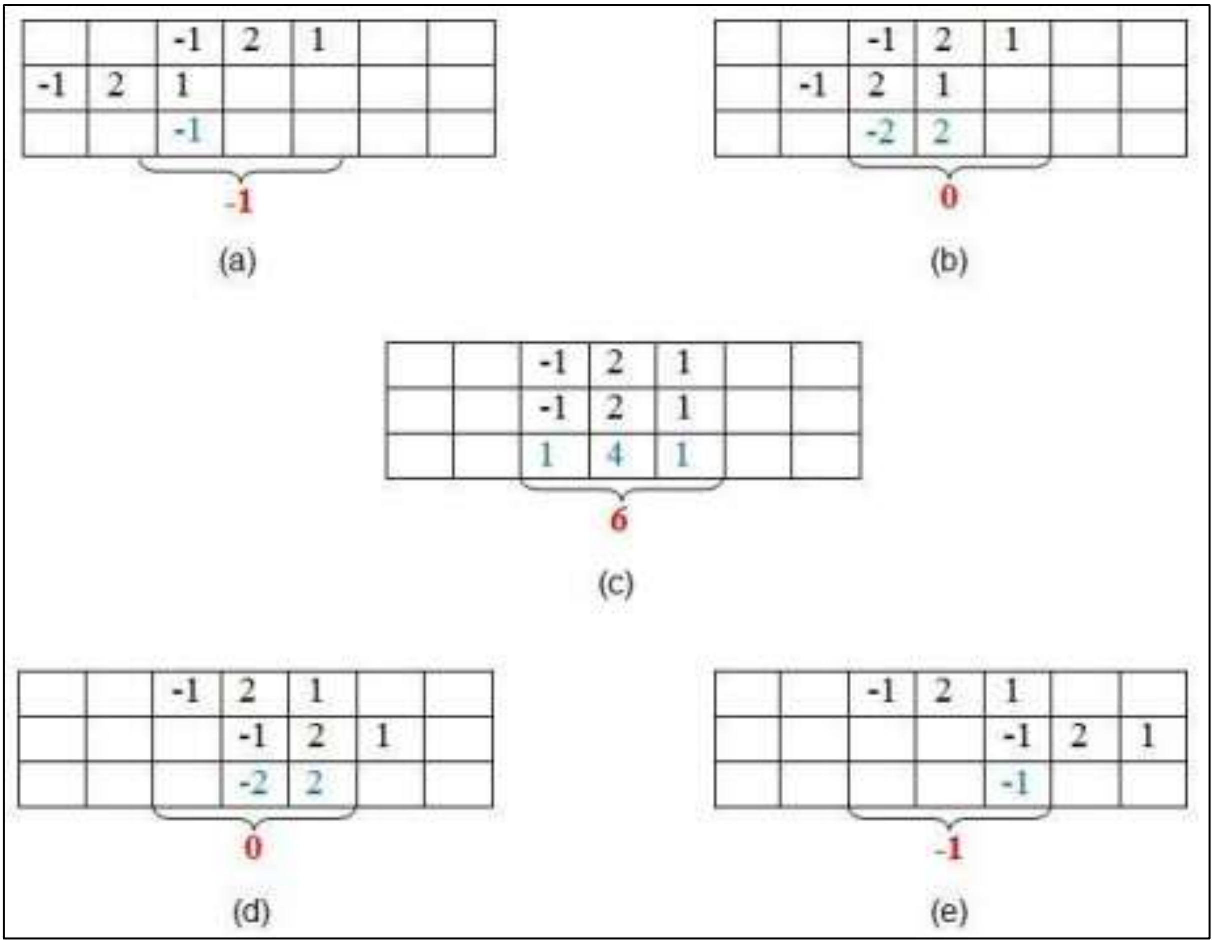
R xx ( τ ) = ∫ ∞ − ∞ x (t) x⋆ (t −τ )dt $R_{xx}(\tau)=\int_{-\infty}^{\infty}x(t)x^{\star}(t-\tau)dt$

where ⋆ ⋆ denotes the complex conjugate.
Similarly the autocorrelation of the discrete time signal *x*[*n*] is expressed as

R xx [ m ] = ∞ ∑ n = − ∞ x [ n ] x⋆ [ n−m ] $R_{xx}[m]=\sum_{n=-\infty}^{\infty}x[n]x^{\star}[n-m]$

Next, the autocorrelation of any given signal can also be computed by resorting to graphical technique. The procedure involves sliding the time-shifted version of the given signal upon itself while computing the samples at every interval. That is, if the given signal is digital, then we shift the given signal by one sample every time and overlap it with the original signal. While doing so, for every shift and overlap, we perform multiply and add.

For example, autocorrelation of the digital signal *x* [<u>*n*</u>] = {-1, 2, 1} can be computed as shown in Figure 1.



***<u>Figure 1: Graphical method of finding autocorrelation</u>***

Here, the first set of samples (those in the first row of every table) refers to the given signal. The second set (in the second row of every table) refers to the samples of its time-shifted version. Next, the samples shown in red color in the third row are obtained by multiplying the corresponding samples of the first two rows.

Finally, we add the samples in the last row of the sample (contained within the curly brackets) so as to obtain the samples of the auto-correlated signal.

Thus, here we find that the samples of the autocorrelated signal Rxx are {-1, 0, 6, 0, -1}, where 6 is the zeroth sample.

The example presented shows that the sample of the autocorrelated signal will be at its maximum value when the overlapping signal best matches the given signal. In this case, it happens when time-shift is zero.

**Cross-Correlation**

This is a kind of correlation, in which the signal in-hand is correlated with another signal so as to know how much resemblance exists_between_them. Mathematical expression for the cross correlation of continuous time signals *x* (*t*) and *y* (*t*) is given by

$$R_{xy}(\tau) = \int_{-\infty}^{\infty} x(t)\, y^{\star}(t-\tau)\, dt$$

Similarly, the cross-correlation of the discrete time signals *x* [*n*] and *y* [*n*] is expressed

as $$R_{xy}[m] = \sum_{n=-\infty}^{\infty} x[n]\, y^{\star}[n-m]$$

Next, just as is the case with autocorrelation, cross-correlation of any two given signals can be found via graphical techniques. Here, one signal is slid upon the other while computing the samples at every interval. That is, in the case of digital signals, one signal is shifted by one sample to the right each time, at which point the sum of the product of the overlapping samples is computed.

For example, cross-correlation of the digital signals x [n] = {-3, 2, -1, 1} and y [n] = {-1, 0, -3, 2} can be computed as shown by Figure 2.

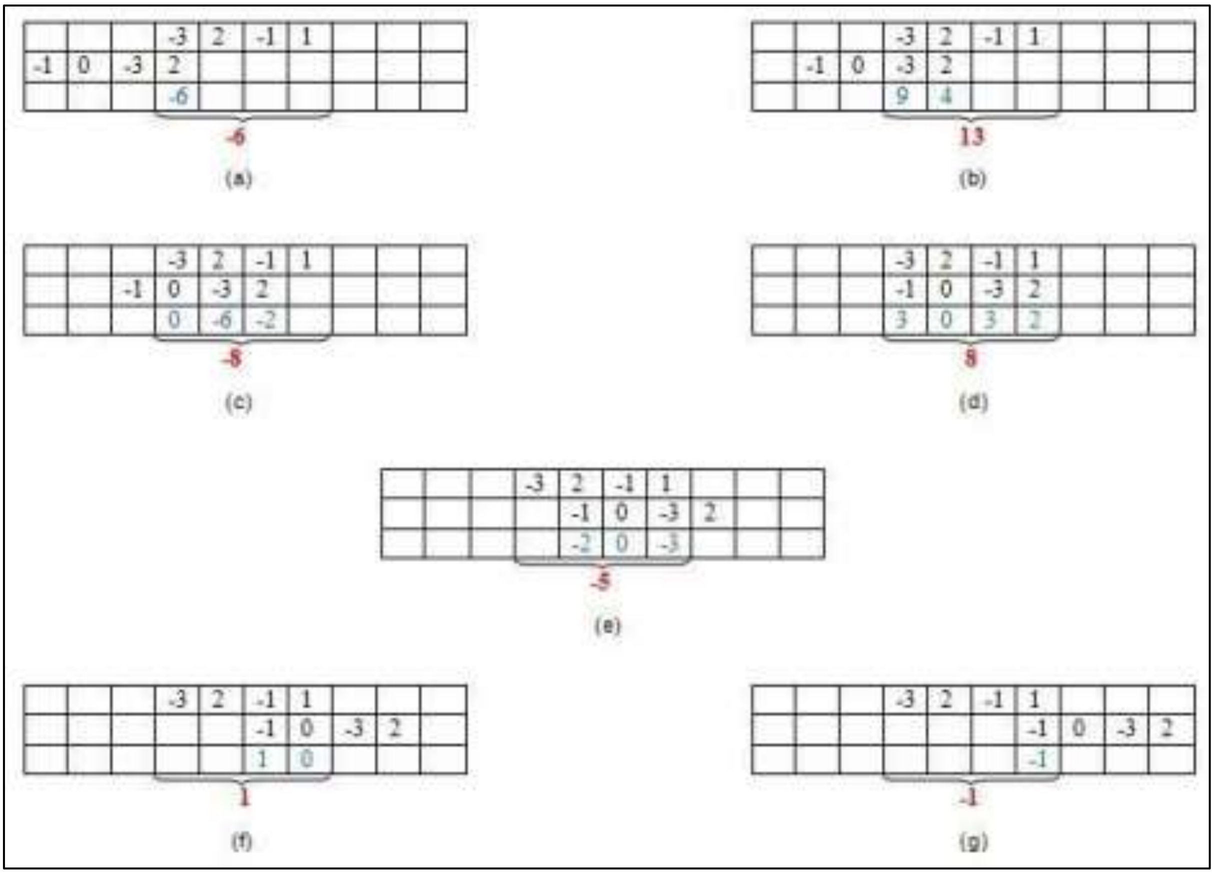



***Figure 2: Graphical method of finding cross-correlation***

Here, the first set of samples (in the first row of every table) refers to the signal x [n] and the second set refers to the samples (in the second row of every table) of the signal y [n].

Next, the samples shown in blue color—those in the third row—are obtained by multiplying the corresponding samples of the first two rows. Finally, we add the samples in the last row (contained within the curly brackets) so as to obtain the samples of the cross-correlated signal.

Thus, here we see that the samples of the cross-correlated signal Rxy are obtained as {-6, 13, -8, 8, -5, 1, -1}, where 8 is the zeroth sample.

Further, the example presented shows that the sample of the cross-correlated signal is at its highest peak, with value 13, when the last two samples of *y* [*n*] overlap with the first two samples of *x* [*n*]. This is because, in this case, the second signal overlaps with the first at its best, as the two samples in each of the signals are identical.

Hence, it can be concluded that the cross-correlation reaches its maximum when the two signals considered become most similar to each other.

**Applications**

As we've seen in the above examples, correlation is useful in real-world scenarios. There are, in fact, many practical applications for correlation. Here are just a few:

1. **Signal processing related to human hearing:** The human ear interprets signals that are *nearly* periodic signals to be *exactly* periodic. This is just like the case where an autocorrelated signal exhibits slightly different maxima-values at regular intervals of time.
2. **Vocal processing:** Correlation can help to determine the tempo or pitch associated with musical signals. The reason is the fact that the autocorrelation can effectively be used to identify repetitive patterns in any given signal.
3. **Determining synchronization pulses:** The synchronization pulses in a received signal, which in turn facilitates the process of data retrieval at the receiver's end. This is because the correlation of the known synchronization pulses with the incoming signal exhibits peaks when the sync pulses are received in it. This point can then be used by the receiver as a point of reference, which makes the system understand that the part of the signal following from then on (until another peak is obtained in the correlated signal indicating the presence of sync pulse) contains data.
4. **Radar engineering:** Correlation can help determine the presence of a target and its range from the radar unit. When a target is present, the signal sent by the radar is scattered by it and bounced back to the transmitter antenna after being highly attenuated and corrupted by noise. If there is no target, then the signal received will be just noise. Now, if we correlate the arriving signal with the signal sent, and if we obtain a peak at a certain point, then we can conclude that a target is present. Moreover, by knowing the time-delay (indicated by the time-instant at which the correlated signal exhibits a peak) between the sent and received signals, we can even determine the distance between the target and the radar.
5. **Interpreting digital communications through noise:** As demonstrated above, correlation can aid in digital communications by retrieving the bits when a received signal is corrupted heavily by noise. Here, the receiver correlates the received signal with two standard signals which indicate the level of '0' and '1', respectively. Now, if the signal highly correlates with the standard signal which indicates the level of '1' more than with the one which represents '0', then it means that the received bit is '1' (or vice versa).
6. **Impulse response identification:** As demonstrated above, cross-correlation of a system's output with its input results in its impulse response, provided the input is zero mean unit variance white Gaussian noise.
7. **Image processing:** Correlation can help eliminate the effects of varying lighting which results in brightness variation of an image. Usually this is achieved by cross correlating the image with a definite template wherein the considered image is searched for the matching portions when compared to a

template (template matching). This is further found to aid the processes like facial recognition, medical imaging, navigation of mobile robots, etc.

8. **Linear prediction algorithms:** In prediction algorithms, correlation can help guess the next sample arriving in order to facilitate the compression of signals.
9. **Machine learning:** Correlation is used in branches of machine learning, such as in pattern recognition based on correlation clustering algorithms. Here, data points are grouped into clusters based on their similarity, which can be obtained by their correlation.
10. **SONAR:** Correlation can be used in applications such as water traffic monitoring. This is based on the fact that the correlation of the signals received by various shells will have different time-delays and thus their distance from the point of reference can be found more easily.

In addition to these, correlation is also exploited to study the effect of noise on the signals, to analyze the fractal patterns, to characterize ultrafast laser pulses, and in many more cases.

**Conclusion:**

Thus we have successfully implemented discrete correlation.

**Code:**

```
clear;

clc;

x=[1 3 -2 1 2 -1 4 4 2];

disp(x,'x');

y=[2 -1 4 1 -2 3];

disp(y,'y');

//Cross corelation rxy[n]:


rxy=convol(x,mtlb_fliplr(y));

disp(rxy,'The Cross-Corelation Operation of the Inputs is
=')
```

**Output:**

```
x

    1.     3.   - 2.     1.     2.   - 1.     4.     4.     2.
```

```
 y
    2.  - 1.    4.    1.  - 2.    3.
 The Cross-Corelation Operation of the Inputs is =
        column  1 to 11
    3.    7.  - 11.    14.    13.  - 15.    28.    6.  -
2.    21.    12.
        column 12 to 14
    12.    6.    4.
```

**Code:**

```
//discrete auto correlation and cross correlation
x=[2 5 0 4];
h=[3 1 4];
x1=x(length(x):-1:1)
h1=h(length(h):-1:1)
rxhn=convol(x,h1)
rhxn=convol(x1,h)
rhxn1=rhxn(length(rhxn):-1:1)
//we observe that rhxn1=rxhn
x=[3 1 -4];
x1=x(length(x):-1:1)
rxxn=convol(x,x1)
//we observe that rxxn is even symmetric about origin
```

**Output:**

```
x1  =
    4.    0.    5.    2.
 h1  =
    4.    1.    3.
 rxhn  =
```

```
   8.     22.     11.     31.     4.     12.
rhxn  =
   12.     4.     31.     11.     22.     8.
rhxn1  =
   8.     22.     11.     31.     4.     12.
x1  =
 - 4.    1.    3.
rxxn  =
 - 12.  - 1.    26.  - 1.  - 12.
```