

CPE 381: Fundamentals of Signals and Systems for Computer Engineers

09 Discrete Signals and Systems

Rahul Bhadani

Electrical & Computer Engineering, The University of Alabama in Huntsville

Outline

1. Discrete-time Signals
2. Operations on Discrete-time Signals
3. Basic Discrete-Time Signals
4. Discrete-time Systems and Their Properties
5. Linear And Non-Linear Filtering of Discrete Signals
6. Two-Dimensional Discrete-Time Signals
7. Two-Dimensional Discrete-Time Systems

$\downarrow \frac{1}{9}$

Discrete-time Signals

Discrete-time Signals

A discrete-time signal $x[n]$ can be thought of as a real- or complex-valued function of the integer sample index n :

$$x[.] : \mathcal{I} \rightarrow \mathcal{R}(\mathcal{C})$$
$$n \mapsto x[n].$$

Example

$$x(t) = 3 \cos\left(2\pi t + \frac{\pi}{4}\right), \quad -\infty < t < \infty$$

$$T_s \leq \frac{\pi}{\Omega_{\max}} = \frac{\pi}{2\pi} = 0.5s/sample$$

Then its discrete version is

$$x[n] = 3 \cos\left(2\pi t + \frac{\pi}{4}\right)|_{t=0.5n} = 3 \cos\left(\pi n + \frac{\pi}{4}\right), \quad -\infty < t < \infty$$

Periodic Discrete-time Signal

$$x[n + kN] = x[n]$$

k : any integer

N : Fundamental period of $x[n]$

Aperiodic discrete-time signals don't satisfy the above property.

Sampling Analog Sinusoid

When sampling an analog sinusoid,

$$x(t) = A \cos(\Omega_0 t + \theta), \quad -\infty < t < \infty$$

of fundamental period $T_0 = 2\pi/\Omega_0$, $\Omega_0 > 0$, we obtain a **periodic discrete sinusoid**.

$$x[n] = A \cos(\Omega_0 T_s n + \theta) = A \cos\left(\frac{2\pi T_s}{T_0} n + \theta\right)$$

provided that

$$\frac{T_s}{T_0} = \frac{m}{N}$$

for the positive integers N and m which are not divisible by each other. To avoid frequency aliasing, the sampling period should also satisfy the Nyquist sampling condition,

$$T_s \leq \frac{\pi}{\Omega_0} = \frac{T_0}{2}$$

Finite-energy, Finite-power Discrete-time Signal

⚡ **Energy:** $\varepsilon_x = \sum_{n=-\infty}^{\infty} |x[n]|^2$

⚡ **Power:** $P_x = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^N |x[n]|^2$

⚡ $x[n]$ is said to have **finite energy** or to be **square summable** if $\varepsilon_x < \infty$.

⚡ $x[n]$ is called **absolutely summable** if

$$\sum_{n=-\infty}^{\infty} |x[n]| < \infty$$

⚡ $x[n]$ is said to have finite power if $P_x < \infty$.

Example

$$x(t) = \begin{cases} 2 \cos(\Omega_0 t - \pi/4) & t \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

- ⚡ Determine its discrete-time signal. Choose $T_s = 0.1$.
- ⚡ Determine if this discrete-time signal has finite energy, finite power and compare these characteristics with those of the continuous-time signal for $\Omega_0 = \pi$ and $\Omega_0 = 3.2$ rad/s.

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation

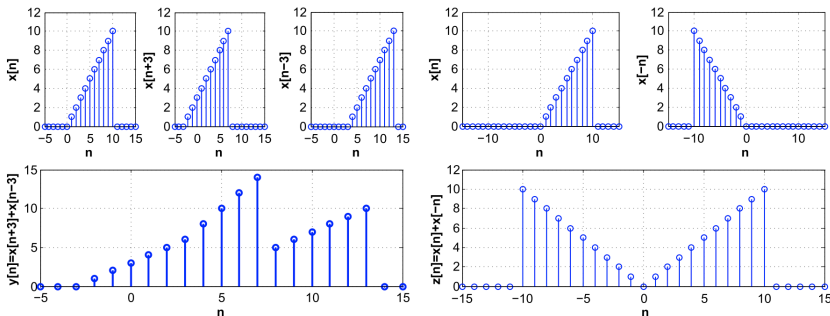


Operations on Discrete-time Signals

Discrete-time Signal Manipulation

A discrete-time signal $x[n]$ is said to be

- ⚡ delayed by L (an integer) samples if $x[n - L]$ is $x[n]$ shifted to the right L samples,
- ⚡ advanced by M (an integer) samples if $x[n + M]$ is $x[n]$ shifted to the left M samples,
- ⚡ reflected if the variable n in $x[n]$ is negated, i.e., $x[-n]$.



Even and Odd Discrete-Time Signal

⚡ $x[n]$ is an **even** signal if $x[n] = x[-n]$.

⚡ $x[n]$ is an **odd** signal if $x[n] = -x[-n]$.

Decomposing a signal into add and even signal:

$$x[n] = \frac{1}{2}(x[n] + x[-n]) + \frac{1}{2}(x[n] - x[-n])$$

where $x_e[n] = \frac{1}{2}(x[n] + x[-n])$ is the even signal component and $x_o[n] = \frac{1}{2}(x[n] - x[-n])$ is the odd signal component.



Basic Discrete-Time Signals

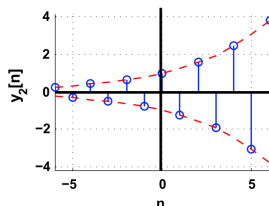
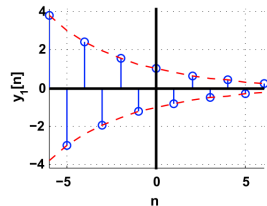
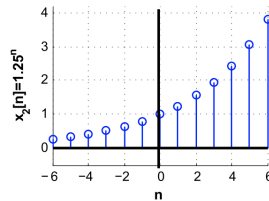
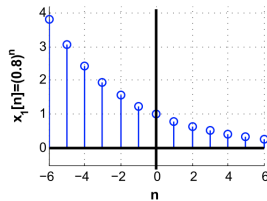
Discrete-Time Complex Exponential

Given a complex number $A = |A|e^{j\theta}$ and $\alpha = |\alpha|e^{j\omega_0}$, a discrete-time complex exponential is a signal

$$\begin{aligned}x[n] &= A\alpha^n = |A||\alpha|^n e^{j(\omega_0 n + \theta)} \\&= |A||\alpha|^n [\cos(\omega_0 n + \theta) + j \sin(\omega_0 n + \theta)]\end{aligned}$$

where ω_0 is a discrete frequency in radians.

Notice that ω will represent discrete frequencies in our discussion, while Ω is used for the continuous frequencies.



Real exponential $x_1[n] = 0.8^n$, $x_2[n] = 1.25^n$ (top)

and

Modulated exponential $y_1[n] = x_1[n] \cos(\pi n)$ and $y_2[n] = x_2[n] \cos(\pi n)$ (bottom).

Discrete-time Sinusoids

A special case of discrete-complex exponential:

⚡ $\alpha = e^{j\omega_0}$

⚡ $x[n] = A\alpha^n = |A|e^{j(\omega_0 n + \theta)} = |A| \cos(\omega_0 n + \theta) + j|A| \sin(\omega_0 n + \theta)$

⚡ The real part of $x[n]$ is a cosine, while the imaginary part is a sine.

⚡ Discrete sinusoids of amplitude A and phase shift θ are periodic if

$$A \cos(\omega_0 n + \theta) = A \sin(\omega_0 n + \theta + \pi/2), \quad -\infty < n < \infty$$

⚡ $\omega_0 = 2\pi m/N$ (rad) is the discrete frequency, for integers m and $N > 0$ which are not divisible. Otherwise, discrete-time sinusoids are not periodic.

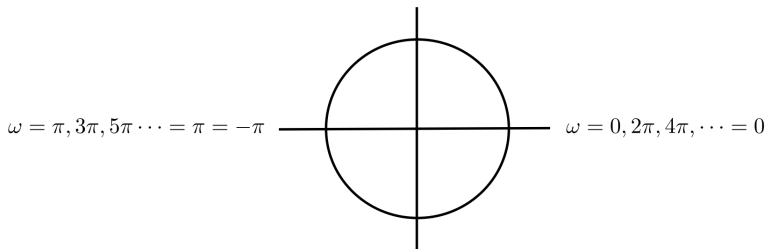
⚡ $\omega_0 + 2\pi k = \omega_0$, where k is an integer.

Limiting range for Discrete Frequencies

To avoid ambiguity in discrete-frequency values, we limit its range

$$-\pi < \omega \leq \pi$$

$$\omega = \pi/2, 5\pi/2, 9\pi/2 \cdots = \pi/2$$



$$\omega = 3\pi/2, 7\pi/2, 11\pi/2, \cdots = -\pi/2$$

Discrete-Time Unit-Step and Unit-Sample Signals

⚡ The unit-step $u[n]$ and the unit-sample $\delta[n]$ discrete-time signals are defined as

$$u[n] = \begin{cases} 1, & n \geq 0, \\ 0, & n < 0 \end{cases}$$
$$\delta[n] = \begin{cases} 1, & n = 0, \\ 0, & \text{otherwise} \end{cases}$$

⚡ $\delta[n] = u[n] - u[n - 1]$

⚡ $u[n] = \sum_{k=0}^{\infty} \delta[n - k] = \sum_{m=-\infty}^n \delta[m]$

Note: $u[n]$ and $\delta[n]$ are NOT sampled versions of the continuous signals $u(t)$ and $\delta(t)$. $u[n]$ and $\delta[n]$ are entirely different signals.

Discrete-Ramp Functions

$$\text{⚡ } r[n] = nu[n]$$

$$\text{⚡ } r[n] = \sum_{k=0}^{\infty} k\delta[n-k] = \sum_{k=0}^{\infty} u[n-k]$$

Generic Representation of Discrete-Time Signals

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n - k]$$

Sifting property of the unit-sample signal

$$\text{⚡ } x[n]\delta[n - n_0] = x[n_0]\delta[n - n_0]$$

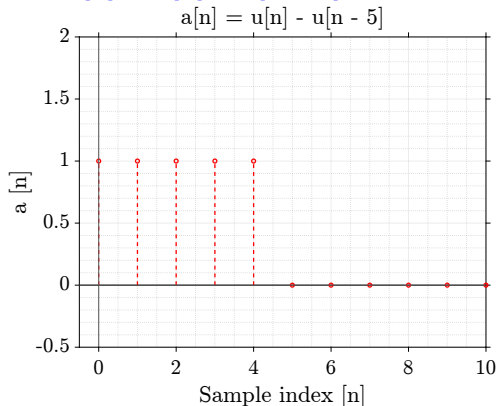
$$\text{⚡ } \delta[n - n_0] = \begin{cases} 1 & n = n_0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{⚡ } x[n] = \cdots + x[-1]\delta[n + 1] + x[0]\delta[n] + x[1]\delta[n - 1] + \cdots = \sum_{k=-\infty}^{\infty} x[k]\delta[n - k]$$

Plotting Discrete Function

```
f = figure(1);
f.Position = [ 286 , 129, 1086 , 833];
n = -10:10;
% Define the unit-step function
u = @(n) (n >= 0);
% Calculate the discrete function
a = u(n) - u(n - 5);
stem(n, a, 'Color', 'red', 'LineWidth', ...
     2, 'LineStyle', '--');
hold on;
%x-axis
line([0 0], [-10, 10], 'Color', ...
     '#555555', 'LineWidth', 1);
%y-axis
line([-10, 10], [0 0], 'Color', ...
     '#555555', 'LineWidth', 1);
xlabel('Sample index [n]', 'Interpreter','latex');
ylabel('a [n]', 'Interpreter','latex');
title('a[n] = u[n] - u[n - 5]', 'Interpreter','latex');
```

Plot $a[n] = u[n] - u[n - 5]$.



https://github.com/rahulbhadani/CPE381_FA24/blob/master/Code/Discrete_Function_Plot.m



Discrete-time Systems and Their Properties

Discrete-time Systems Representation

A discrete-time system is a transformation of a discrete-time input signal $x[n]$ into a discrete-time output signal $y[n]$

$$y[n] = \mathcal{S}\{x[n]\}$$

Linearity

⚡ **Scaling:** $\mathcal{S}\{ax[n]\} = a\mathcal{S}\{x[n]\}$

⚡ **Additivity:** $\mathcal{S}\{x[n] + v[n]\} = \mathcal{S}\{x[n]\} + \mathcal{S}\{v[n]\}$

⚡ Equivalently, superposition applies:

$$\mathcal{S}\{ax[n] + bv[n]\} = a\mathcal{S}\{x[n]\} + b\mathcal{S}\{v[n]\}$$

Time-invariance

- ⚡ If for an input $x[n]$ the corresponding output is $y[n] = \mathcal{S}\{x[n]\}$, the output corresponding to an advanced or a delayed version of $x[n]$, $x[n \pm M]$, for an integer M , is $y[nM] = \mathcal{S}\{x[n \pm M]\}$, or the same as before but shifted as the input.
- ⚡ In other words, the system is not changing with time.

Recursive And Non-Recursive Discrete-Time Systems

⚡ Input: $x[n]$

⚡ Output: $y[n]$

⚡ Recursive System (infinite impulse response (IIR) system):

$$y[n] = - \sum_{k=1}^{N-1} a_k y[n-k] + \sum_{m=0}^{M-1} b_m x[n-m], \quad n \geq 0$$

⚡ Non-Recursive System (Finite impulse response (FIR) system):

$$y[n] = \sum_{m=0}^{M-1} b_m x[n-m]$$

These two equations shown above are called difference equations.

Autoregressive Discrete System

$$y[n] = ay[n - 1] + bx[n], \quad n \geq 0$$

with initial condition of $y[-1]$ given.

Example: Autoregressive Moving-Average System

Consider a system represented by the difference-equation:

$$y[n] = 0.5y[n-1] + x[n] + x[n-1]n \geq 0, y[-1]$$

Consider two cases:

- ⚡ Let the initial condition be $y[-1] = -2$, and the input $x[n] = u[n]$ first and then $x[n] = 2u[n]$. Find the corresponding outputs.
- ⚡ Let the initial condition be $y[-1] = 0$, and the input $x[n] = u[n]$ first and then $x[n] = 2u[n]$. Find the corresponding outputs.

Use the above results to determine in each case if the system is linear. Find the steady-state response, i.e., $\lim_{n \rightarrow \infty} y[n]$.

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation

Dynamic Discrete-Time Systems Represented By Difference Equations

A recursive discrete-time system is represented by a difference equation

$$y[n] = - \sum_{k=1}^{N-1} a_k y[n-k] + \sum_{m=0}^{M-1} b_m x[n-m], \quad n \geq 0$$

with the initial condition given as $y[-k]$ for $k = 1, \dots, N-1$.

This difference equation could be the approximation of an ordinary differential equation representing a continuous-time system being processed discretely.

Zero-input and Zero-state Responses

Just as in the continuous-time case, the system being represented by the difference equation is not LTI unless the initial conditions are zero and the input is causal.

The complete response of a system represented by the difference equation can be shown to be composed of zero-input and zero-state responses,

$$y[n] = y_{zi}[n] + y_{zs}[n]$$

The component $y_{zi}[n]$ is the response when the input $x[n]$ is set to zero, thus it is completely due to the initial conditions.

The response $y_{zs}[n]$ is due to the input only, as we set the initial conditions to zero.

Convolution Sum

⚡ Remember the generic representation of the signal: $x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n - k]$

⚡ The Convolution Sum gives the output of the LTI system:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n - k] = \sum_{m=-\infty}^{\infty} x[n - m]h[m]$$

Example

Consider an autoregressive system represented by a first-order difference equation

$$y[n] = 0.5y[n-1] + x[n], \quad n \geq 0.$$

Find the impulse response $h[n]$ of the system and then compute the response of the system to $x[n] = u[n] - u[n-3]$ using the convolution sum.

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation



Linear And Non-Linear Filtering of Discrete Signals

Linear Filtering

In general, the filters under consideration are linear and shift-invariant. Thus, the output such as signals or images are characterized by the convolution sum between the input signal and the filter impulse response.

Example: Averaging Filter

Let $y[n] = x[n] + \eta[n]$

⚡ $x[n]$: The averaging filter.

⚡ $\eta[n]$: Gaussian noise.

Averaging filter of M -th order:

$$z[n] = \frac{1}{M} \sum_{k=0}^{M-1} y[n - k]$$

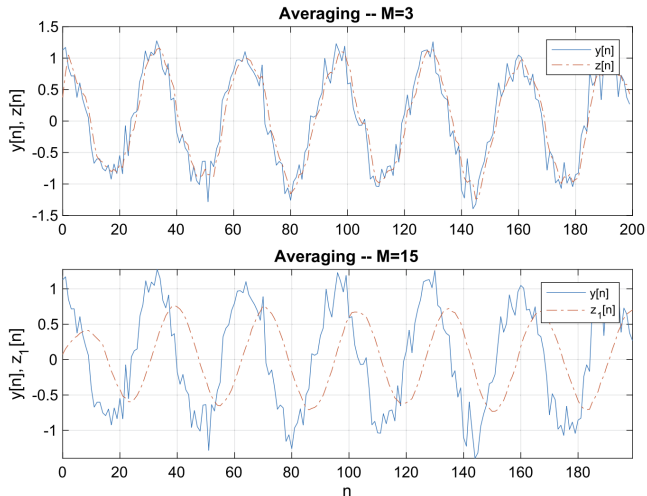
Note that higher-order filters will have a larger delay.

Averaging Filter in MATLAB

```
N=200;n=0:N-1;  
x=cos(pi*n/16); % input signal  
noise=0.2*randn(1,N); % noise  
y=x+noise; % noisy signal  
% averaging linear filter with M=3  
z=averager(3,y);  
% averaging linear filter with M=15  
z1=averager(15,y);
```

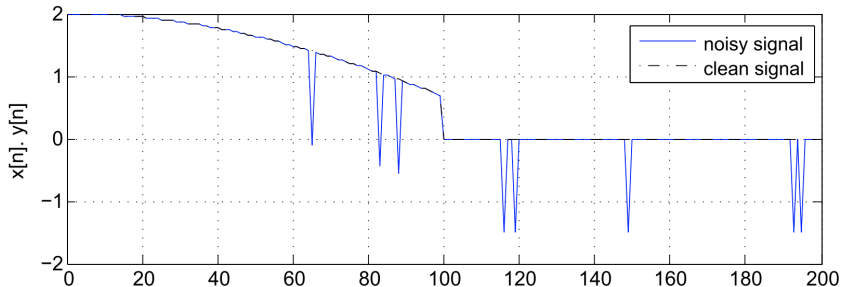
```
function y=averager(M,x)  
% Moving average of signal x  
% M: order of averager  
% x: input signal  
%  
b=(1/M)*ones(1,M);  
y=filter(b,1,x);
```

Averaging Filter in MATLAB



Nonlinear Filtering

Not all linear filters are capable of removing noise, such as impulsive noise. In such a case, we can use non-linear filtering methods, such as median filters.



Median Filtering

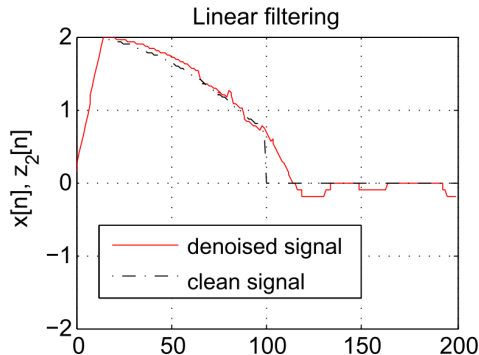
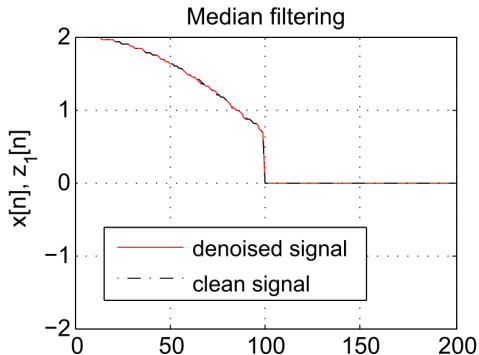
A median filter considers a certain number of samples (the example shows a 5th-order median filter), orders them according to their values, and chooses the one in the middle (i.e., the median) as the filter's output. Such a filter is non-linear as it does not satisfy superposition.

Median Filter in MATLAB

```
clear all; clf
N=200;n=0:N-1;
% impulsive noise
for m=1:N,
    d=rand(1,1);
    if d>=0.95,
        noise(m)=-1.5;
    else
        noise(m)=0 ;
    end
end
```

```
x=[2*cos(pi*n(1:100)/256) zeros(1,100)];
y1=x+noise;
% linear filtering
z2=averager(15,y1);
% non-linear filtering -- median filtering
z1(1)=median([0 0 y1(1) y1(2) y1(3)]);
z1(2)=median([0 y1(1) y1(2) y1(3) y1(4)]);
z1(N-1)=median([y1(N-3) y1(N-2) y1(N-1) y1(N) 0]);
z1(N)=median([y1(N-2) y1(N-1) y1(N) 0 0]);
for k=3:N-2,
    z1(k)=median([y1(k-2) y1(k-1) y1(k) y1(k+1) y1(k+2)]);
end
```

Median Filter vs Averaging Filter



Causality of a Discrete-Time LTI System

A discrete-time system \mathcal{S} is causal if:

- ⚡ whenever the input $x[n] = 0$, and there are no initial conditions, the output is $y[n] = 0$,
- ⚡ the present output $y[n]$ does not depend on future inputs.

An LTI system can be noncausal, such is the case of the following LTI system that computes the moving average of the input:

$$y[n] = \frac{1}{3}(x[n+1] + x[n] + x[n-1])$$

Causality of a Discrete-Time LTI System

- ⚡ An LTI discrete-time system is causal if the impulse response of the system is such that $h[n] = 0$ for $n < 0$.
- ⚡ A signal $x[n]$ is said to be causal if $x[n] = 0$ for $n < 0$.
- ⚡ For a causal LTI discrete-time system with a causal input $x[n]$ its output $y[n]$ is given by

$$y[n] = \sum_{k=0}^n x[k]h[n-k], \quad n \geq 0$$

where the lower limit of the sum depends on the input causality, $x[k] = 0$ for $k < 0$, and the upper limit on the causality of the system, $h[n-k] = 0$ for $n-k < 0$ or $k > n$.

Bounded Input–Bounded Output (BIBO) Stability

- ⚡ Stability characterizes useful systems.
- ⚡ A stable system provides well-behaved outputs for well-behaved inputs.
- ⚡ Bounded input–bounded output (BIBO) stability establishes that for a bounded (which is what is meant by ‘well-behaved’) input $x[n]$ the output of a BIBO stable system $y[n]$ is also bounded.
- ⚡ Hence, if there is a finite bound $M < \infty$ such that $|x[n]| < M$ for all n (you can think of it as an envelope $[-M, M]$ inside which the input $x[n]$ is) the output is also bounded, i.e., $|y[n]| < L$ for $L < \infty$ and all n .

Bounded Input–Bounded Output (BIBO) Stability

⚡ An LTI discrete-time system is said to be BIBO stable if its impulse response $h[n]$ is absolutely summable

$$\sum_k |h[k]| < \infty$$

⚡ Or,

$$|y[n]| \leq \left| \sum_{k=-\infty}^{\infty} x[n-k]h[k] \right| \leq |x[n-k]| |h[k]| \leq M \sum_{k=-\infty}^{\infty} |h[k]| \leq MN < \infty$$

provided that $\sum_{k=-\infty}^{\infty} |h[k]| < N < \infty$, or that the impulse response be absolutely summable.

Consider $L = MN$.

Example

Consider an autoregressive system $y[n] = 0.5y[n - 1] + x[n]$.
Determine if the system is BIBO stable

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation



Two-Dimensional Discrete-Time Signals

Two-dimensional discrete signals

A discrete two-dimensional signal $x[m, n]$ is a mapping of integers $[m, n]$ into real values that is not defined for non-integer values.

Two-dimensional Impulse Signal

$$\delta[m, n] = \begin{cases} 1, & [m, n] = [0, 0] \\ 0, & [m, n] \neq [0, 0] \end{cases}$$

A signal $x[m, n]$ defined in a support $[M_1, N_1] \times [M_2, N_2]$, $M_1 < M_2$, $N_1 < N_2$ can be written as

$$x[m, n] = \sum_{k=M_1}^{M_2} \sum_{\ell=N_1}^{N_2} x[k, \ell] \delta[m - k, n - \ell]$$

Two-dimensional Unit-step Signal

Two-dimensional unit-step signal $u_1[m, n]$, with support in the first quadrant

$$u_1[m, n] = \begin{cases} 1, & m \geq 0, n \geq 0, \\ 0, & \text{otherwise} \end{cases} = \sum_{k=0}^{\infty} \sum_{\ell=0}^{\infty} \delta[m - k, n - \ell]$$

Two-dimensional Unit-ramp Signal

A two-dimensional unit-ramp signal $r_1[m, n]$, with support in the first quadrant

$$r_1[m, n] = \begin{cases} mn, & m \geq 0, n \geq 0, \\ 0, & \text{otherwise} \end{cases} = \sum_{k=0}^{\infty} \sum_{\ell=0}^{\infty} k\ell \delta[m - k, n - \ell]$$

Separable Signals

A class of two-dimensional signals of interest are separable signals $y[m, n]$ that are the product of two one-dimensional signals, one being a function of m and the other of n

$$y[m, n] = y_1[m]y_2[n]$$

$\delta[m, n] = \delta[m]\delta[n]$ is separable.



Two-Dimensional Discrete-Time Systems

Two-dimensional System

A two-dimensional system is an operator \mathcal{S} that maps an input $x[m, n]$ into a unique output $y[m, n] = \mathcal{S}(x[m, n])$.

We only consider the LTI 2-D system:

⚡ Linearity: $\mathcal{S}\left(\sum_{i=1}^I a_i x_i[m, n]\right) = \sum_{i=1}^I a_i \mathcal{S}(x_i[m, n]) = \sum_{i=1}^I a_i y_i[m, n]$

⚡ Shift-Invariance: $\mathcal{S}(x_i[m - M, n - N]) = y_i[m - M, n - N]$

A system satisfying these two conditions is called **linear shift-invariant** or **LSI**.

Impulse Response in LSI Two-dimensional System

Suppose then the input $x[m, n]$ of an LTI system and that the response of the system to $\delta[m, n]$ is $h[m, n]$ or the impulse response of the system. Then,

$$\begin{aligned} y[m, n] &= \sum_k \sum_\ell x[k, \ell] \mathcal{S}(\delta[m - k, n - \ell]) \\ &= \sum_k \sum_\ell x[k, \ell] h[m - k, n - \ell] = (x * h)[m, n] \end{aligned}$$

This is also called a 2-d Convolution Sum.

Separable LSI Systems

An LSI system is **separable** if its impulse response $h[m, n]$ is a separable sequence, i.e., $h[m, n] = h_1[m]h_2[n]$. Then we can write the convolution sum as

$$\begin{aligned} y[m, n] &= \sum_{k=-\infty}^{\infty} \sum_{\ell=-\infty}^{\infty} x[k, \ell] h_1[m - k] h_2[n - \ell] \\ &= \sum_{k=-\infty}^{\infty} h_1[m - k] \left[\sum_{\ell=-\infty}^{\infty} x[k, \ell] h_2[n - \ell] \right] \end{aligned}$$

Notice that the term in the bracket is the convolution of the sum of $h_2[n]$ and the input for fixed values of k .

Separable LSI Systems

Let

$$y_1[k, n] = \sum_{\ell=-\infty}^{\infty} x[k, \ell] h_2[n - \ell] = \sum_{\ell=0}^{\infty} x[k, \ell] h_2[n - \ell]$$

then,

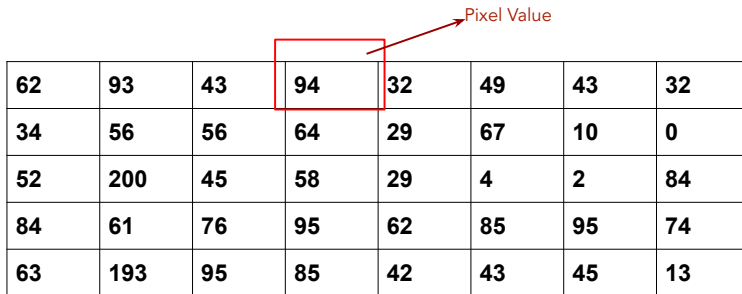
$$y[m, n] = \sum_{k=-\infty}^{\infty} y_1[k, n] h_1[m - k] = \sum_{k=0}^{\infty} y_1[k, n] h_1[m - k]$$

which is one-d convolution sum of $h_1[m]$ and $y_1[k, n]$ for fixed values of n .

The final forms above are obtained using that both input and impulse response are supported in the first quadrant.

Images as 2-dimensional Discrete Signals

Represented as a matrix of integer values



Pixel Value

| | | | | | | | |
|----|-----|----|----|----|----|----|----|
| 62 | 93 | 43 | 94 | 32 | 49 | 43 | 32 |
| 34 | 56 | 56 | 64 | 29 | 67 | 10 | 0 |
| 52 | 200 | 45 | 58 | 29 | 4 | 2 | 84 |
| 84 | 61 | 76 | 95 | 62 | 85 | 95 | 74 |
| 63 | 193 | 95 | 85 | 42 | 43 | 45 | 13 |

Filtering on Images

Form a new image whose pixels are modified version of original pixel values.

Goals of filtering:

- ⚡ Extract useful information from the images
 - Features (edges, corners, blobs. . .)
- ⚡ Modify or enhance image properties
 - super-resolution; in-painting; de-noising

2D discrete-space systems (filters)

$$f[n, m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n, m]$$

$$g = \mathcal{S}[f], \quad g[n, m] = \mathcal{S}\{f[n, m]\}$$

$$f[n, m] \xrightarrow{\mathcal{S}} g[n, m]$$

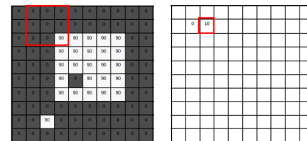
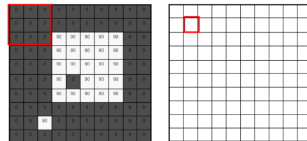
2D Filter Example

2D discrete-space moving average over a 3×3 window of a neighborhood

$$g[n, m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{\ell=m-1}^{m+1} f[k, \ell] = \frac{1}{9} \sum_{k=-1}^1 \sum_{\ell=-1}^1 f[n-k, m-\ell]$$

Or, $(f * h)[m, n] = \frac{1}{9} \sum_{k, \ell} f[k, \ell] h[m-k, n-\ell]$

$$\frac{1}{9} \begin{matrix} & \text{n} \\ \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix} \end{matrix}$$



Example

Consider a separable impulse response

$$\begin{aligned}h[m, n] &= \begin{cases} 1, & 0 \leq m \leq 1, 0 \leq n \leq 1 \\ 0, & \text{otherwise} \end{cases} \\&= (u[m] - u[m - 2])(u[n] - u[n - 2]) = h_1[m]h_2[n]\end{aligned}$$

For an input

$$x[m, n] = \begin{cases} 1, & 0 \leq m \leq 1, 0 \leq n \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

find the output of the system $y[m, n]$.

Blank space for calculation

Blank space for calculation

Blank space for calculation

Blank space for calculation