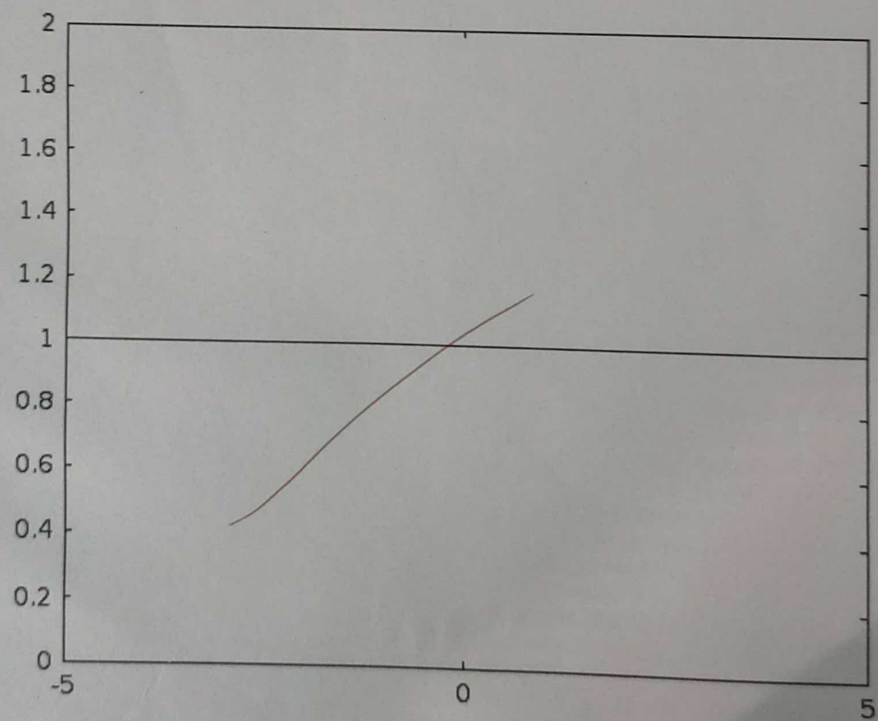
```
% Experiment 1 To plot the spectrum of a pluse width 10.
% ADITYA KUMAR
% 2021UCA1824
clc;
clear all;
close all;
syms t w
x=1;
expw=exp(-1*1i*w*t);
z=int(x*expw,t,-5,5);
xlabel('t')
ylabel('x(t)')
figure(1);
fplot('1',[-5 5])
figure(2);
fplot(z)
```

*Warning: fplot will not accept character vector or string inputs in a future release. Use fplot(@(t)1*ones(size(t))) instead.*



AIM: To verify plot the spectrum of a pulse of width 10.

Material Required : MATLAB Software

Theory: The given pulse is a rectangular pulse. The fourier Transform of $x(t)$ is to be determined.

$$x(t) = \begin{cases} 1 & |t| < T \\ 0 & |t| > T \end{cases}$$

The fourier transform of $x(t)$ is given as:-

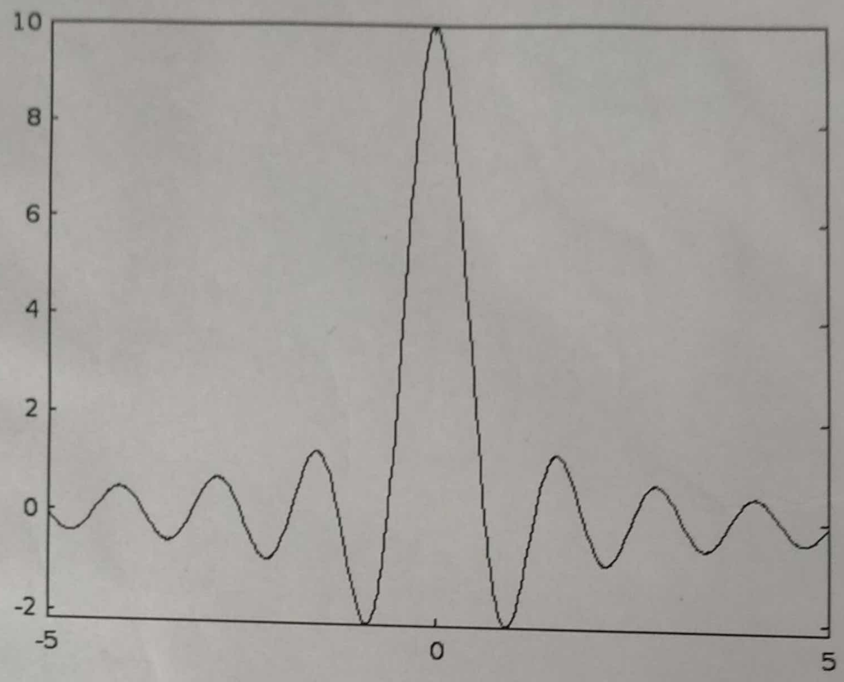
$$x(i\omega) = \int_{-\infty}^{\infty} x(t)\, e^{-i\omega t} \cdot dt = \int_{-T}^{T} e^{-i\omega t} \cdot dt$$

$$= \left[\frac{e^{-i\omega t}}{-i\omega} \right]_{-T}^{T} = \frac{-1}{i\omega} \left[e^{-i\omega T} - e^{i\omega T} \right]$$

$$= \frac{1}{i\omega} \left[e^{i\omega T} - e^{-i\omega T} \right] = \frac{2i}{i\omega} \left[\frac{e^{i\omega T} - e^{-i\omega T}}{2i} \right]$$

$$= \frac{2}{\omega} \left[\sin \omega T \right]$$

$$\Rightarrow \boxed{x(i\omega) = \frac{2 \sin \omega T}{\omega}}$$



Published with MATLAB® R2022b

Magnitude spectrum wrt frequency:- $x(i\omega)=0$ when $\sin\omega T=0$

$\therefore \omega T = n\pi \Rightarrow \omega = \frac{n\pi}{T}$ $(n=0, \pm1, \pm2, \cdots)$

At, $\omega T = 0$, $x(i\omega) = \lim_{\omega\to 0} \frac{2\sin \omega T}{\omega}$

$\Rightarrow x(i\omega) = 2T$ at $\omega = 0$

Conclusion:- The magnitude spectrum with respect is plotted for a rectangular pulse of width 10.

```
% Experiment 2 2. To verify following properties of Fourier Transform:
% i. Time Shifting
% ADITYA KUMAR
% 2021UCA1824
clc;
clear all;
close all;
syms t w
x=cos(t);
t0=2;
xt0=cos(t-t0);
Left=fourier(xt0,w)
X=fourier(x,w);
Right=exp(-j*w*t0)*X
```

```
Left =

pi*(dirac(w - 1)*exp(-2i) + dirac(w + 1)*exp(2i))


Right =

pi*exp(-w*2i)*(dirac(w - 1) + dirac(w + 1))
```

Published with MATLAB® R2022b

AIM: To verify the following properties of Fourier Transform.

(i.) Time shifting
(ii.) Frequency shifting
(iii.) convolutional

Material Required: MATLAB software.

Theory:

(i.) Time shifting: For a time domain signal
$x(t) \overset{F}{\rightleftharpoons} X(i\omega)$

Time ~~-iωt~~ shifting, $x(t \pm t_0) = x(i\omega t) \cdot e^{\pm j\omega t_0}$

~~$x(i\omega) = \int_{-\infty}^{\infty} x(t) \cdot e^{-i\omega t} \cdot dt$~~

replacing t with $(t-t_0)$, $x(t) \rightarrow x(t-t_0) \rightleftharpoons x'(i\omega)$

$$x'(i\omega) = \int_{-\infty}^{\infty} x(t-t_0) \cdot e^{-i\omega t} \cdot dt$$

$t - t_0 = \tau \Rightarrow t = \tau + t_0$
$dt = d\tau$

$$x'(i\omega) = \int_{-\infty}^{\infty} x(\tau) \cdot e^{-i\omega(\tau + t_0)} \cdot d\tau$$

$$= \int_{-\infty}^{\infty} x(\tau) \cdot e^{-i\omega\tau} \cdot e^{-i\omega t_0} \cdot d\tau$$

$$x'(i\omega) = e^{-i\omega t_0} \int_{-\infty}^{\infty} x(\tau) \cdot e^{-i\omega\tau} \cdot d\tau = e^{-j\omega \tau} \cdot x(i\omega)$$

$\overset{F}{\rightleftharpoons} x(t - t_0)$

$(\because t = \tau)$

$$\equiv e^{\pm j\omega t_0} \cdot x(i\omega) \overset{F}{\rightleftharpoons} x(t \pm t_0)$$

Q.E.D

```
% Experiment 2 2. To verify following properties of Fourier Transform;
% iii. Convolutional
% ADITYA KUMAR
% 2021UCA1824
clc;
clear all;
close all;
% Define two signals
x = [1 2 3 4];
h = [1 1 1 1];

% Compute the convolution
y = conv(x,h);

% Compute the Fourier transform of each signal
X = fft(x);
H = fft(h);

% Compute the Fourier transform of the convolution
Y = X .* H;

% Compute the inverse Fourier transform to obtain the convolution in time
 domain
y_verify = ifft(Y);

% Compare the result
disp(y);
disp(y_verify);

     1     3     6    10     9     7     4

    10    10    10    10
```

Published with MATLAB® R2022b

(ii.) Frequency shifting :- Consider $x(t) \rightleftharpoons x(j\omega)$

To prove: $e^{\pm j\omega t} \cdot x(t) \rightleftharpoons x[j(\omega \mp \omega_0)]$

$$x(j\omega) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j\omega t} \cdot dt$$

$$e^{+j\omega_0 t} \cdot x(t) \rightleftharpoons x'(j\omega)$$

$$x'(j\omega) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j(\omega-\omega_0)t} \cdot dt$$

$$\therefore x'(j\omega) = x[j(\omega-\omega_0)] \rightleftharpoons e^{+j\omega_0 t} \cdot x(t)$$

Q.E.D

(iii.) Convolution in Time :- Let $x_1(t) \rightleftharpoons x_1(j\omega)$
$x_2(t) \rightleftharpoons x_2(j\omega)$

To prove: $x_1(t) * x_2(t)$ [Convolution] $\rightleftharpoons x_1(j\omega) \cdot x_2(j\omega)$

$x(t) \rightleftharpoons x(j\omega) \Rightarrow x(j\omega) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j\omega t} \cdot dt$

(or)

$$x(j\omega) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x_1(t) \cdot x_2(t-t_0) \cdot d\tau) \cdot e^{-j\omega t} \cdot dt$$

$$t - \tau = \lambda \Rightarrow t = \lambda + \tau$$

$$\therefore dt = d\lambda$$

$$x(j\omega) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_1(\tau) \cdot x_2(\lambda) \cdot d\tau \cdot e^{-j\omega\lambda} \cdot e^{-j\omega\tau} \cdot d\lambda$$

$$x(j\omega) = \int_{-\infty}^{\infty} x_1(\tau) \cdot e^{-j\omega\tau} \, d\tau \cdot \int_{-\infty}^{\infty} x_2(\lambda) \cdot e^{-j\omega\lambda} \cdot d\lambda$$

$$= x_1(j\omega) \cdot x_2(j\omega)$$

```
% Experiment 2 2. To verify following properties of Fourier Transform:
% ii. Frequency Shifting
% ADITYA KUMAR
% 2021UCA1824
clc;
clear all;
close all;
syms t w
x=t^3;
w0=3;
Le=exp(j*w0*t)*x;
Left=fourier(Le,w)


Left =

-pi*dirac(3, w - 3)*2i
```

Published with MATLAB® R2022b

Experiment :

Date

$\therefore X(j\omega) = X_1(j\omega) \cdot X_2(j\omega) \rightleftharpoons x(t) = x_1(t) \cdot x_2(t)$

Q.E.D.

Conclusion:- The properties of time shifting, frequency shifting, and convolution in time have been verified for Fourier Transformation.

```
%experiment 3: To generate uniform random number and plot its density
%function. Find its mean and variance.
% Aditya Kumar
% 2021UCA1824

clc;
clear;
clear all;

N=100000;
x=rand(1,N);mux=mean(x);sigmax2=var(x);
step=0.1;
b=1;a=0;
range1=a:step:b;
range = -3:step:3;

f = 1/b-a;

plot(range1,f,'b*');hold("on");
h=hist(x,range);
step=0.1;
simf= h/(step.*sum(h));
plot(range,simf,'m-')
xlabel('range')
ylabel('PDF')
title('PDF of Uniform Distributed Random Variabe')
legend('theorectical','Simulated')
```

Aim: To generate uniform random number and plot its density function. Also find its mean and variance.

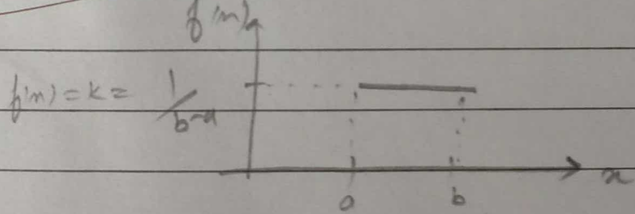
Material Required: MATLAB software.

Theory: A continuous random variable x is said to be follow continuous uniform/Rectangular distribution on interval (a,b) if its probability Distribution function is given by,

$$f(x) = \begin{cases} k ; & a < x < b \\ 0 ; & \text{otherwise} \end{cases}$$

$$\int_{-\infty}^{\infty} f(x) \cdot dx = 1 \qquad \therefore \int_a^b k \cdot dx = 1 \qquad \therefore k = \frac{1}{(b-a)}$$

i.e; $f(x) \begin{cases} 1/b-a ; & a<x<b \\ 0 ; & \text{otherwise} \end{cases}$

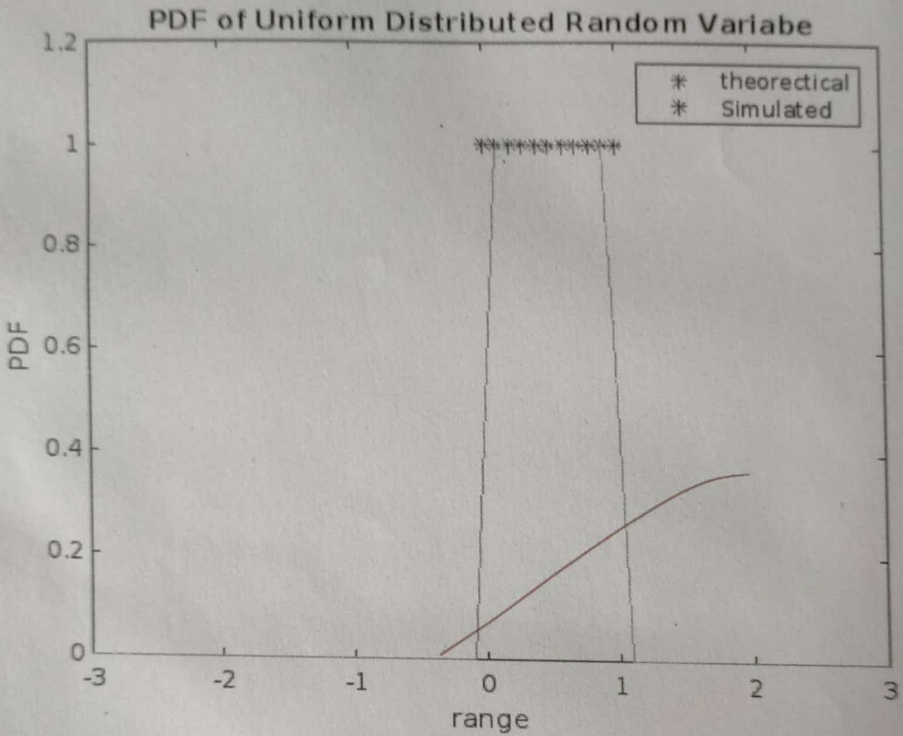


Mean: $E(x) = \int_{-\infty}^{\infty} x \cdot f(x) \cdot dx$ (or) $\int_a^b x\left(\frac{1}{b-a}\right) dx$

$$= \frac{1}{(b-a)} \left[\frac{x^2}{2}\right]_a^b = \frac{1}{(b-a)} \frac{(b^2-a^2)}{2} = \frac{(a+b)}{2}$$

Variance :- $Var(x) = E(x^2) - (E(x))^2$

$$E(x^2) = \int_{-\infty}^{\infty} x^2 \cdot f(x) \cdot dx = \int_a^b \frac{x^2 \cdot 1 \cdot dx}{(b-a)} = \frac{1}{(b-a)} \left[\frac{x^3}{3}\right]_a^b$$



Published with MATLAB® R2022b

Experiment :

Date

Page No. 14

$$= \frac{b^3-a^3}{3(b-a)} = \frac{(a^2+b^2+ab)}{3}$$

$$E(x) = \frac{a+b}{2}$$

$$Var(x) = \frac{a^2+b^2+ab}{3} - \frac{(a+b)^2}{4}$$

$$= \frac{4b^2+4a^2+4ab-3a^2-6ab-3b^2}{12}$$

$$= \frac{a^2+b^2-2ab}{12} = \frac{(a-b)^2}{12}$$

Conclusion:- Rectangular Distribution function is ~~period~~ plotted for certain range, and its mean and variance is calculated.

DELTA®

```
% exp 4: to generate gaussian distributed random number and plot its
% density function. Find its mean and variance.
% Aditya Kumar
% 2021UCA1824

clc;
clear all;
close all;
N=100000;
x=randn(1,N);
mux=mean(x);
sigmax2=var(x);
step=0.1;
range=-3:step:3;
F= 1/sqrt(2*pi*sigmax2).*exp(-(range-mux).^2./(2*sigmax2));
figure(1);
plot(range,F,'b-*'), hold on;
h=hist(x, range);
step=0.1;
simF=h/(step.*sum(h));
plot(range, simF,'m-');
xlabel('range');
ylabel('plot');
title("Gaussian distribution");
legend('Theoritical','Simulation');
```

Aim: To generate Gaussian Random number and plot its density function. Also find its mean and variable.

Material Required:- MATLAB Software

Theory:- A continous random variable with the probability Distribution function: $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$; $-\infty < x < \infty$; $-\infty < \mu < \infty$; $\sigma > 0$

Mean: $\int_{-\infty}^{\infty} x f(x) \cdot dx = \int_{-\infty}^{\infty} \frac{x \cdot 1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \cdot dx$

$= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} x \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \cdot dx$; Let $\frac{x-\mu}{\sigma} = z$

$\therefore dz = \sigma dz$

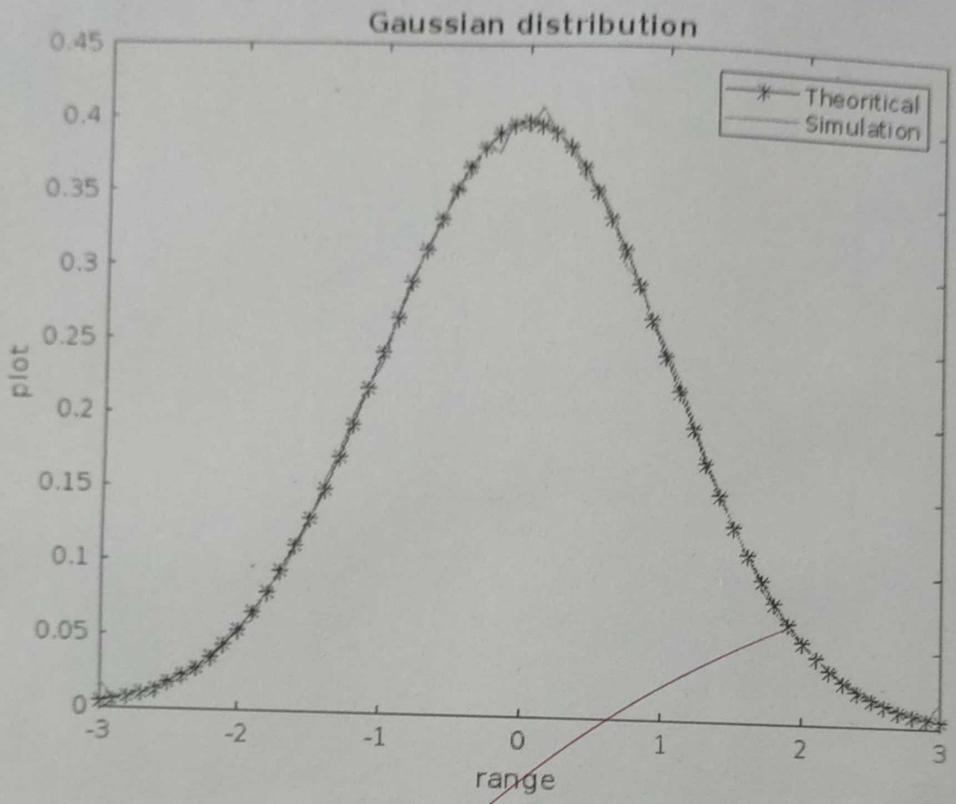
$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} (\mu + \sigma z)\, e^{-z^2/2} \cdot dz$

$= \frac{1}{\sqrt{2\pi}} \left[\int_{-\infty}^{\infty} \mu \cdot e^{-z^2/2} dz + \underbrace{\int_{-\infty}^{\infty} \sigma z \cdot e^{-z^2/2} dz}_{\to 0} \right]$ $\because$ odd function

$= \frac{1}{\sqrt{2\pi}} \times 2\mu \int_{0}^{\infty} e^{-z^2/2} \cdot dz$ [$\because$ even function]

$E(x) = \frac{2\mu}{\sqrt{2\pi}} \int_{0}^{\infty} e^{-p} \frac{\cdot dp}{\sqrt{2p}}$ [Let $\frac{z^2}{2} = p$, $z \cdot dz = dp$]

$= \frac{2}{\sqrt{2\pi} \times \sqrt{2}} \int_{0}^{\infty} p^{-1/2} \cdot e^{-p} \cdot dp = \frac{\mu}{\sqrt{\pi}} \times \frac{1}{\sqrt{2}}$ $= \frac{\mu}{\sqrt{\pi}} \times \sqrt{\pi}$



Published with MATLAB® R2022b

$$\boxed{E(x) = \mu}$$

$$E(x^2) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} x^2 e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \cdot dx$$

$$= \frac{\sqrt{2}\,\sigma}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} (\sqrt{2}\,\sigma t + \mu)^2 \cdot e^{-t^2} dt \qquad \left[\frac{x-\mu}{\sigma} = t \quad \therefore dx = \sigma dt\right]$$

$$= \frac{1}{\sqrt{\pi}}\left(2\sigma^2 \int_{-\infty}^{\infty} t^2 \cdot e^{-t^2} dt + 2\sqrt{2}\,\sigma\mu \int_{-\infty}^{\infty} t \cdot e^{-t^2} dt + \mu^2 \int_{-\infty}^{\infty} e^{-t^2} \cdot dt\right)$$

$$= \frac{1}{\sqrt{\pi}}\left(2\sigma^2 \int_{-\infty}^{\infty} t^2 \cdot e^{-t^2} \cdot dt + 2\sqrt{2}\,\sigma\mu \left[\frac{-1}{2} e^{-t^2}\right]_{-\infty}^{\infty} + \mu^2\sqrt{\pi}\right)$$

$$= \frac{1}{\sqrt{\pi}}\left(2\sigma^2 \int_{-\infty}^{\infty} t^2 \cdot e^{-t^2} dt + 2\sqrt{2} \cdot \sigma\mu \cdot 0\right) + \mu^2$$

$$= \frac{2\sigma^2}{\sqrt{\pi}} \int_{-\infty}^{\infty} t^2 \cdot e^{-t^2} dt = \frac{2\sigma^2}{\sqrt{\pi}}\left(\left[\frac{-t}{2} e^{-t^2}\right]_{-\infty}^{\infty} + \frac{1}{2}\int_{-\infty}^{\infty} e^{-t^2} \cdot dt\right)$$

$$= \frac{2\sigma^2}{\sqrt{\pi}} \times \frac{1}{2} \int_{-\infty}^{\infty} e^{-t^2} \cdot dt \qquad = \frac{2\sigma^2\sqrt{\pi}}{2\sqrt{\pi}} = \sigma^2$$

$$\boxed{Var(x) = E(x^2) - (E(x))^2}$$

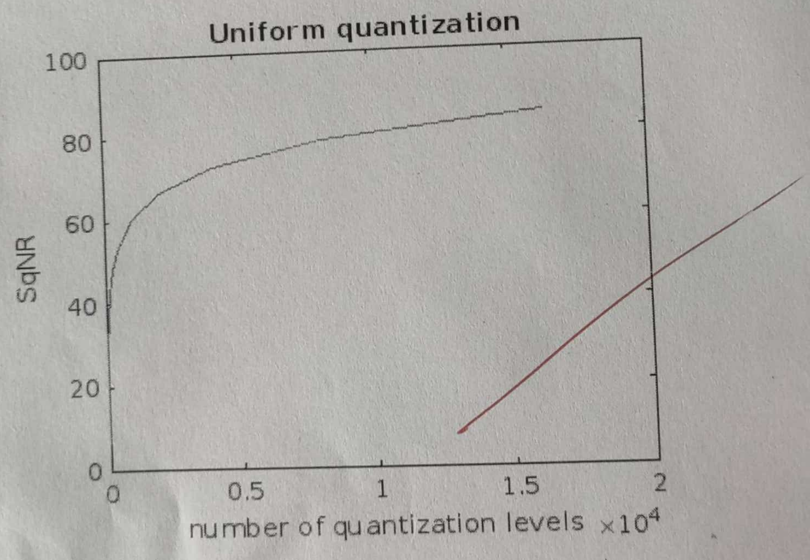
$$= \boxed{\sigma^2 - \mu^2}$$

Experiment :

Date

Conclusion: Gaussian distribution function is plotted for a given range, and their mean and variance are constant.

```
% Experiment 5: Compute the Signal to quantization Noise ratio of Uniform
% Quantization. Plot SNQR versus Quantization levels.
% Aditya Kumar
% 2021UCA1824
clc;clear all;close all;
n=100; x=rand(1, n); vmin=min(x); vmax=max(x); xpow=sum (x.^2)/n;
for i=1:1:14
 L(i) =2^i;
 d= (vmax-vmin)/L (i);
 for j=1:length(x)
 start =min(x) ;
 while (start<x(j))
 start=start+d;
 end
 xq(j)=start-d;
 if (start==x(j))
 xq(j)=start;
 end
 end
 err=x-xq;
 noisepow(i)=sum(err.^2)/n;
end
sqnr=xpow./noisepow;
sqnrdb=10. *log10(sqnr);
plot (L,sqnrdb)
xlabel ('number of quantization levels');
ylabel ('SqNR');
title ('Uniform quantization')
```



Published with MATLAB® R2023a

Aim: Compute the signal to Quantisation Noise ratio of uniform quantisation. Plot SNQR vs quantisation levels.

Material Required: Matlab Software

Theory: Signal to quantisation noise ratio (SQNR) of uniform quantisation depends upon the no. of bits used for quantisation, and independent of amplitude of the input signal.
Assuming input signal has maximum amplitude A, and is uniformly distributed over range (-A, A), quantiser has step size Δ.

$$SNR = 20\log\left[\frac{\text{Signal rms voltage}}{\text{Noise rms voltage}}\right]$$

$$\overline{q_{e}^{2}} = \frac{1}{\Delta}\int_{-\Delta/2}^{\Delta/2} q^2\,dq = \frac{1}{\Delta}\left[\frac{q^3}{3}\right]_{-\Delta/2}^{\Delta/2} = \frac{1}{3\Delta}\left[\frac{2\Delta^2}{8}\right] = \frac{\Delta^2}{12}$$

rms noise $\Rightarrow q_e = \Delta/2\sqrt{3}$

~~full scale signal; $V_{max} - V_{min} = V_{fs}$~~

peak ~~voltage~~ $= V_{fs}/2$

rms peak voltage $= V_{fs}/2\sqrt{2}$

$$\Delta = \frac{V_{max} - V_{min}}{2^n} = \frac{V_{fs}}{2^n} \Rightarrow V_{fs} = \Delta \cdot 2^n.$$

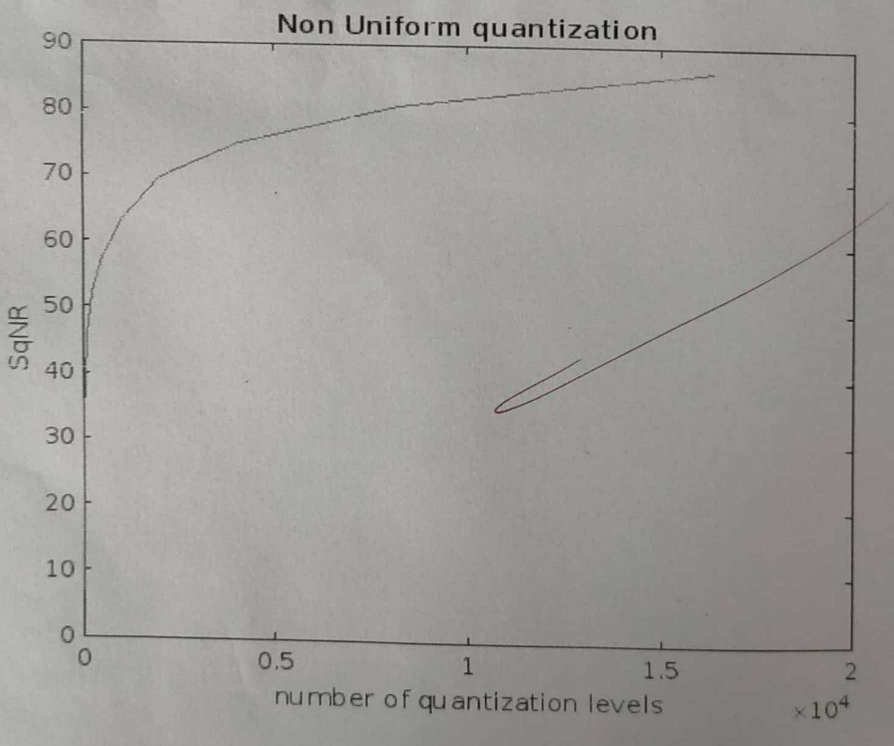
$$\therefore \text{signal rms} = \frac{\Delta \cdot 2^n}{2\sqrt{2}}$$

$$\therefore SNR = 20\log\left(\frac{2^n \cdot \Delta}{2\sqrt{2}} \Big/ \frac{\Delta}{2\sqrt{3}}\right) = 20\log(2^n) + 20\log(\sqrt{3/2})$$

$$\boxed{SNR = 6.02n + 1.76} \text{ dB}$$

Conclusion: SQNR versus quantisation plot is plotted for a uniform quantisation.

```
% Experiment 6: Compute the Signal to quantization Noise ratio of Non-Uniform
% Quantization.Plot SNQR versus Quantization levels.
% Aditya Kumar
% 2021UCA1824
clc; clear all;close all;
n=100; x=rand (1,n) ;
u=255;
xcomp= ((log(1+(abs (x)./max(x)).*u))./log (1+u) ) ; %compressed sample
vmin=min (xcomp); vmax=max (xcomp) ; xpow=sum (xcomp.^2)/n;
for i=1:1:14
 L(i)=2^i;
 d= (vmax-vmin)/L(i) ;
 for j=1:length (xcomp)
 start =min (xcomp);
 while (start<xcomp (j) )
 start=start+d;
 end
 xq(j)=start-d;
 if(start==x(j))
 xq(j) =start;
 end
 end
 err=xcomp-xq;
 noisepow (i)=sum (err.^2) /n;
end
sqnr=xpow./noisepow;
sqnrdb=10.*log10 (sqnr) ;
plot (L, sqnrdb)
xlabel ('number of quantization levels');
ylabel ('SqNR');
title ('Non Uniform quantization')
```



Published with MATLAB® R2023a

Experiment : 6

Date

~~Aim~~ Aim: Compute the signal to quantisation noise ratio of non uniform quantisation. Plot SQNR versus quantisation levels.

Material Required: Matlab software

Theory: Assume a non uniform quantiser with step size Δ quantisation noise power for non uniform quantisation.

$$Q = \int f(x) \cdot \Delta^2 \cdot dx$$ [$f(x) \rightarrow$ probability density fun of input signal]

signal power, $S = A^2/3$

$$SQNR = 10 \log_{10} (S/Q)$$

$$= 10 \log_{10} \left[\frac{A^2/3}{(\Delta^2 \int f(x) dx)} \right]$$

SQNR of non uniformly quantised signal depends on input signal distribution, quantisation levels, and quantisation step sizes. Hence, it can be determined only when these characteristics are known.

SQNR of non uniformly quantised signal is lower than that of a uniformly quantised signal with the same number of quantisation levels, because more quantisation levels occur in regions with higher probability of signal amplitude occurence, and vice versa.

∴ quantisation errors is smaller in regions of high signal density, but larger in regions of low signal density.

Conclusion: SQNR versus quantisation levels for non uniform quantisation is plotted.

DELTA®

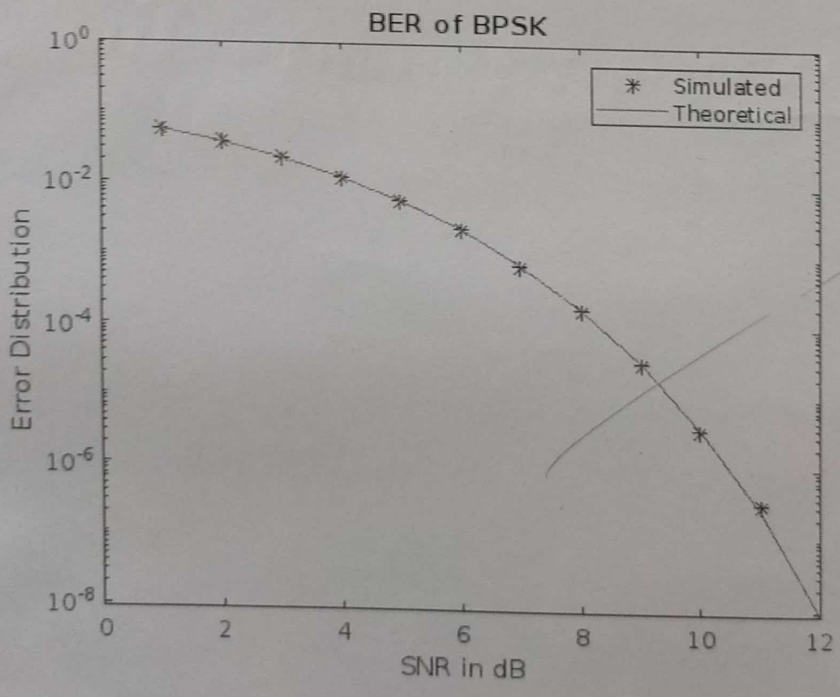
```
% To study passband digtital communication technique BPSK. Calculate the
% BER of BPSK modulated signal.
% Aditya Kumar
% 2021UCA1824

clc;
clear all;
close all;

N = 10^8;
a =rand(1,N) > 0.5;
s = 2.*a-1;
n = (1/sqrt(2))*(randn(1,N) + 1i* rand(1,N));
snr_dB = 1:1:12;
snr_ratio = 10.^(0.1.*snr_dB);
for i=1:length(snr_dB)
    y = 10.^(0.05.*snr_dB(i)).*s + n;
    adec = real(y)>0;
    err(i) = size(find(a-adec),2);
end
sim_avg_err = err/N;
th_avg_err = 0.5 * erfc(sqrt(snr_ratio));

figure(1);
semilogy(snr_dB,sim_avg_err,'b*'); hold on
semilogy(snr_dB,th_avg_err,'m-');

title("BER of BPSK");
legend('Simulated','Theoretical');
xlabel('SNR in dB');
ylabel('Error Distribution');
```



Published with MATLAB® R2023a

Aim: Study of passband digital communication technique. BPSK. Calculate the BER of BPSK modulated signal.

Material required: Matlab Software

Theory: Probability error of PSK

$1 \rightarrow A\cos\omega_c t = x_1(t)$

$0 \rightarrow -A\cos\omega_c t = x_2(t)$

$S(t) = x_1(t) - x_2(t)$

$= 2A\cos\omega_c(t)$

$$V_{max}^2 = \frac{2}{N_0}\int_0^T 4A^2\cos\omega_c t \cdot dt = \frac{4A^2T}{N_0} \qquad \because \left[E_s = \int_0^T (s(t))^2 \cdot dt\right]$$

$$P_E = \frac{1}{2}\,\text{erfc}\left[\frac{P^2_{max}}{8}\right]^{1/2} = \frac{1}{2}\,\text{erfc}\left[\frac{A^2T}{2N_0}\right]^{1/2}$$

$$= \frac{1}{2}\,\text{erfc}\left[\frac{E_s}{N_0}\right]^{1/2}$$

Modulation of BPSK is done using a balance modulator, which multiplies the two signals applied at the input. for a zero binary input, the phase will be 0° and for high input, phase reversal is of 180°.

Conclusion: The Bit error rate of BPSK modulated signal is calculated.

```
%Experiment No. 8:- Generate (7,4) Linear Block Code given a generator matrix
%G=[(1 1 0 1 0 0 0),(0 1 1 0 1 0 0),(1 1 1 0 0 1 0),(1 0 1 0 0 0 1)].
%Assume that the received vector is [1 0 0 1 0 0 1].Find the syndrome.
% Aditya Kumar
% 2021UCA1824
clc;clear all; close all;
G = [1 1 0 1 0 0 0;0 1 1 0 1 0 0;1 1 1 0 0 1 0;1 0 1 0 0 0 1];

r = [1 0 0 1 0 0 1];
[K,N] = size(G);
m = dec2bin(0:(2.^K)-1)-'0';
c = mod((m*G),2);
P=G(:,1:N-K);
I=eye(N-K);
PT=P.';
H = horzcat(PT,I);
HT= H.';
S= mod((r*HT),2);
disp(S);
```

```
     0     1     0
```

Published with MATLAB® R2023a

Aim: Given a linear block code with generator matrix

$$G = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

(a.) calculate no. of valid code words N and code rate R_c, specify complete code set C.
(b.) Determine generator matrix 'G' of appropriate systematic (separable) code 'C''.

Material Required: Matlab software

Theory: Linear code words can be added in modulo 2 arithmetic to produce a third code word in the code. An n bit code word contains 'k' message bits and r check bits $(n = k + r)$.

- Hamming weight (w) → no. of 1's in code word, and $w(C_1 \oplus C_2)$ is the hamming distance, $d(C_1, C_2)$ between a pair of code words.
- The minimum distance (d_{min}) detects upto (s) errors iff $d_{min} \geq s+1$ and correct upto 't' errors if $d_{min} \geq 2t+1$
- Generator matrix, $G = [I_k : P]_{k \times n}$, and given a message word 'm', the set of valid code words, $C = mG$.
- Parity check ~~matrix~~ matrix, $H = [P^T : I_m]$ where, m → length of check bits, $m = n-k$.
  and syndrome code, $S = r[H^T]$, where r → recieved data

Conclusion:- The valid code ~~code~~ words and code ratio (rate) at the given generator matrix was calculated.

```
%Exp 9:To generate a M/M/1 Queue having infinite buffer space with
%parameters (lamda,mu) and plot the average delay per packet vs lamda/mu
%Aditya Kumar
%2021UCA1824
clear ; close all ;clc; n=100000;
 %Generating Lambda(Interarrival Time) And Mu(Service Time)
%Both Are Exponentially Distributed
x=rand(1,n);
ex=(log(1-x)); count=hist(ex,20); lamda=count/(5*n); x=rand(1,n); ex=(log(1-
x)); count=hist(ex,20); mu=count/(n);
ltrue=max(lamda); mutrue=max(mu);
%True Rho Value
rhotrue=ltrue/mutrue;
 %Assigning X-Axis
 xaxis=linspace(0,rhotrue,length(lamda));
%Arrival Time Plot
plot(xaxis,lamda,'-d','Linewidth',1.2);hold on;
%Service Time Plot
plot(xaxis,mu,'-d','Linewidth',1.2);
%Calculating Waiting Time
Wt(1) = lamda(1) + mu(1); for i=2:length(lamda)
if Wt(i-1) > lamda(i)
%extra_waiting_time
ewt=Wt(i-1) - lamda(i);
Wt(i) = ewt + mu(i); elseif Wt(i-1) <= lamda(i)
Wt(i) = mu(i); end
end

%Waiting Time Plot
plot(xaxis,Wt,'-h','Linewidth',1.2)
grid on; hold off;
legend('\lambda','\mu','Waiting time','location','NW'); title('\lambda, \mu
& Wt','fontsize',16)
xlabel('\rho = \lambda / \mu -->>','Fontsize',14,'fontweight','b')
ylabel('average delay per packet','Fontsize',14,'fontweight','b')
```

Aim:- To generate a M/M/1 Queue having infinite buffer space with parameter (λ, μ) and plot the average delay per packets v/s λ/μ.

Material Required:- Matlab software.

Theory:- M/M/1 is a queuing model where a the arrivals follow a poision process with parameters 'λ', arriving in a time interval 't'.

ii) Interval between any two successive arrivals is exponential distributed ~~=~~ with parameter 'λ'.

iii) Time taken to complete a single service is exponentially distributed with parameter μ.

iv) Number of servers is one.

v) Both, population and queue size can be infinity.

vi) Order of service is assumed to be FIFO.

Little's Result relates the numbers ~~of patient~~ packets and waiting time:

$$N = \lambda W$$

$$\text{then, } N = \sum_{i=0}^{\infty} i\phi_i = \sum_{i=0}^{\infty} i\,\rho(1-\rho) = \rho/(1-\rho)$$

$$W = \frac{N}{\lambda} = \frac{1}{\mu(1-\rho)} \; ; \; W_q = W - \frac{1}{\mu} = \frac{\rho}{\mu(1-\rho)}$$

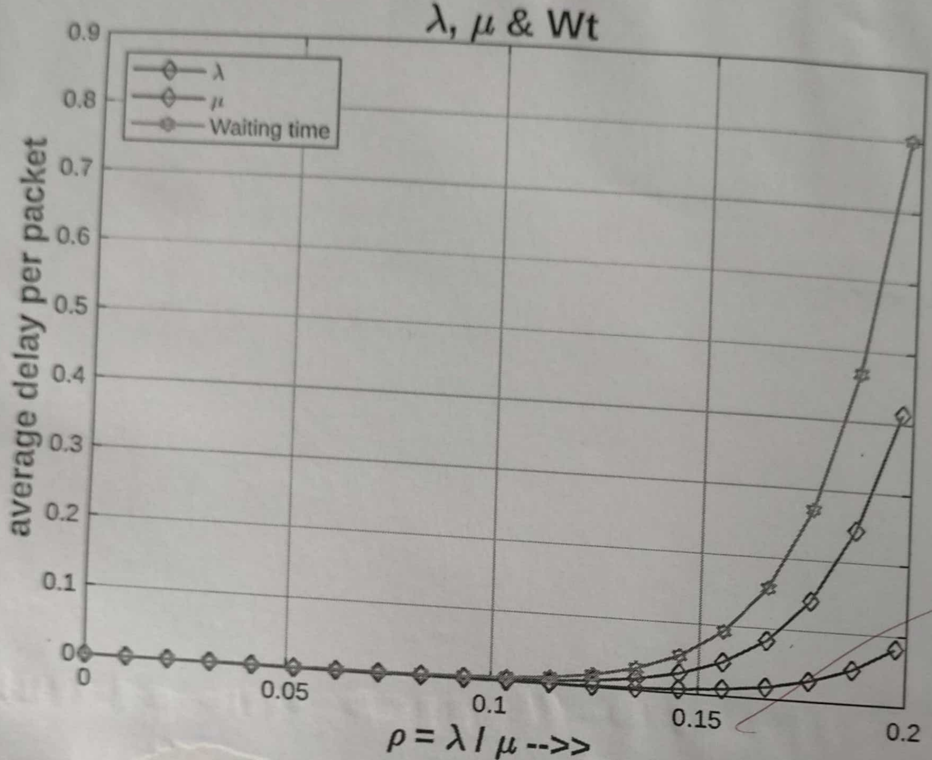
$$N_q = \lambda W_q = \frac{\rho^2}{(1-\rho)} \text{, where}$$

N → no. of packets in system
N_q → no. of packets in queue
W → Waiting time in sequence
W_q → waiting time in queue.

Conclusion:- M/M/1 queue is generated and average delay v/s (λ/μ) was plotted.

DELTA

Ex-9 - Result



λ, μ & Wt
λ
μ
Waiting time
average delay per packet
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
ρ = λ / μ -->>

Aim:- To generate a M/M/1 Queue having infinite buffer space with parameter (λ, μ) and plot the average delay per packet v/s λ/μ.

Software used :- Matlab.

Theory:- The M/M/1 system is made of a poision arrival, one exponential (poision) server, FIFO queue of unlimited capacity and unlimited customer population. Note that these assumptions are very strong, not statisfied for practical system. ~~Neverth~~ Nevertheless the M/M/1 model shows clearly the basic ideas and method of Queuing. Theory. Next two chapter summarise the basic properties of the poision process and give derivation of the M/M/1 theoritical model.

$$P_n = \pi_k = \frac{(1-\rho)\rho^k}{(1-\rho^{k+1})}$$

```
%Exp 10:To generate a M/M/1 Queue having finite buffer space with
% parameters (lamda, mu) and plot blocking probability with respect to
% variation with buffer space.
%Aditya Kumar
%2021UCA1524
lambda = 2; % arrival rate
mu = 3; % service rate
buffer_sizes = 0:20; % vary buffer size from 0 to 20

blocking_probabilities = zeros(size(buffer_sizes)); % preallocate for
efficiency

for i = 1:length(buffer_sizes)
    buffer_size = buffer_sizes(i);
    if buffer_size == 0
        blocking_probabilities(i) = 1 - lambda/mu; % no buffer
    else
        rho = lambda/mu;
        p0 = 1 - rho;
        summation = 0;
        for j = 0:buffer_size
            summation = summation + (rho^j)/factorial(j);
        end
        blocking_probabilities(i) = (rho^buffer_size)/
(factorial(buffer_size)*p0*summation); % compute blocking probability
    end
end

plot(buffer_sizes, blocking_probabilities, 'o-'); % plot blocking
probability vs. buffer size
xlabel('Buffer Size');
ylabel('Blocking Probability');
title(['M/M/1 Queue with Finite Buffer, \lambda = ', num2str(lambda), ', \mu
= ', num2str(mu)]);
```

Experiment :

Date

Page No. 32