**DIGITAL COMMUNICATION – LAB**

**ETEC-357**



**NAME:** SHIVAM SINGH

**ENROLLMENT NO:** 07313302720

**CLASS & SEC:** CSE-5A

**GROUP:** 2

**SUBMITTED TO: SHRISTY GOSWAMI**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **INDEX** | | | | |
| **S.NO** | **AIM** | **DATE OF EXPERIMENT** | **DATE OF**  **SUBMISSION** | **REMARKS** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**SCILAB**

Scilab is a free open-source, cross-platform numerical computation package and a high-level, numerically oriented programming language. It can be used for signal processing, statistical analysis, image enhancement, fluid dynamics simulations, numerical optimization and modelling, simulation of explicit and implicit dynamical systems and symbolic manipulations.

Scilab was created in 1990 by researchers from INRIA and École nationale des ponts et chaussées (ENPC).

Scilab 6.1.1 is released under the terms of the GNU General Public License (GPL) v2.0.

Prior to version 6.0.0, Scilab was previously licensed under the terms of the CeCILL license v2.1 and continues to be available under such terms.

**COMMANDS**

**clc()** : Clear Command Window-clears all input and output from the Console.

**subplot** : subplot(m,n,p) or subplot(mnp) virtually grids the graphics window into an m-by-n matrix of sub-windows and selects the pth sub-window for receiving the forthcoming drawings.

**plot** : plots a set of 2D curves.

**xlabel** : sets or updates the x-axis label or/and its properties

**ylabel** : sets or updates the y-axis label or/and its properties

**title()** : used to set and display a title at the top of the current or given axes, or to change properties of the existing title.

**squarewave(t, percent)** : generates a square wave such that percent is the percent of the period in which the signal is positive.

**plot2d3()** : same as plot2d but curves are plotted using vertical bars. By default, successive plots are superposed.

**set()** : This function can be used to modify the value of a specified property from a graphics entity or a GUI object.

**end** : Used at end of loops or conditionals. for, while, if, select must be terminated by end.

**while** : Opens a block of instructions iterated on a heading condition. While clause must be terminated by end.

**If** : The if statement evaluates a logical expression and executes a group of statements when the expression is true

**else** : Used with if and select.

**length()** : Number of characters of a string. Number of elements of an array or list.

**gca()** : This routine returns the handle of the current axes for the current figure.

**Experiment-1**

**Aim 🡺** To study Sampling Theorem

**Software Used 🡺** Scilab 6.1.1

**Theory 🡺** In signal processing, **sampling** is the reduction of a continuous-time signal to a discrete-time signal. A common example is the conversion of a sound wave to a sequence of samples.

A sample is a value or set of values at a point in time and/or space. When a source generates an analog signal and if that has to be digitized, having 1s and 0s i.e., High or Low, the signal has to be discretized in time.

This discretization of analog signal is called as **Sampling**.

**Sampling frequency** is the reciprocal of the sampling period. This sampling frequency, can be simply called as Sampling rate. The sampling rate denotes the number of samples taken per second, or for a finite set of values.

**Sampling Frequency** = 𝟏

𝑻𝒔 = 𝒇𝒔

where, Ts is sampling period,

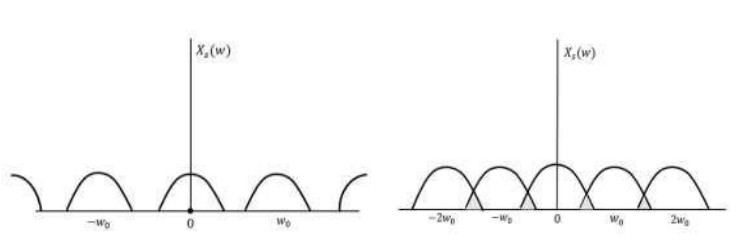
fs is sampling frequency.

For an analog signal to be reconstructed from the digitized signal, the sampling rate should be highly considered. The rate of sampling should be such that the data in the message signal should neither be lost nor it should get overlapped. Hence, a rate was fixed for this, called as **Nyquist rate.**

fs = 2W

This rate of sampling is called as **Nyquist rate.**

**Sampling Theorem**: The sampling theorem states that, “a signal can be exactly reproduced if it is sampled at the rate fs which is greater than twice the maximum frequency W.”



**Code 🡺**

clc;

fm=10e3;

fs=500e3;

ncyc=4;

t=0:(1/fs):(ncyc\*(1/fm));

x=sin(2\*3.14\*fm\*t);

subplot(3,1,1);

plot(t,x);

xlabel("time");

ylabel("amp");

title("message signal");

y=squarewave(2\*3.14\*fm\*t,100);

subplot(3,1,2);

plot2d3(t,y);

xlabel("time");

ylabel("amp");

title("impulse signal");

z=x.\*y;

subplot(3,1,3);

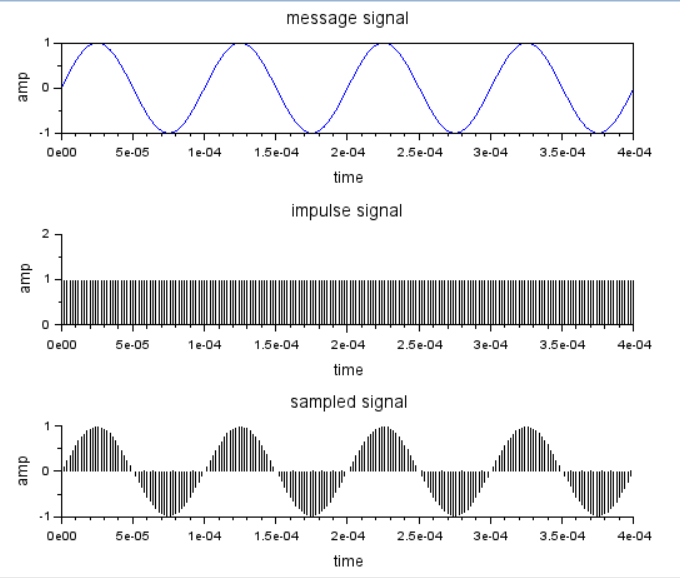
plot2d3(t,z);

xlabel("time");

ylabel("amp");

title("sampled signal");

**Output 🡺**



**Result 🡺**

Sampling has been studied and the sampled signal has been plotted successfully on the graph.

**Experiment-2**

**Aim 🡺 Write a program to study different line encoding techniques**

**Software Used 🡺 Scilab 6.1.1**

**Theory 🡺**

A **line code** is the code used for data transmission of a digital signal over a transmission line. This process of coding is chosen so as to avoid overlap and distortion of signal such as sinter-symbol interference.

## **Properties of Line Coding**

## Following are the properties of line coding −

* As the coding is done to make more bits transmit on a single signal, the bandwidth used is much reduced.
* For a given bandwidth, the power is efficiently used.
* The probability of error is much reduced.
* Error detection is done and the bipolar too has a correction capability.
* Power density is much favorable.
* The timing content is adequate.
* Long strings of **1s** and **0s** is avoided to maintain transparency.

### **Types of Line Coding**

There are 3 types of Line Coding

* Unipolar
* Polar
* Bi-polar

## **Unipolar Signaling**

Unipolar signalling is also called as **On-Off Keying** or simply **OOK**.

The presence of pulse represents a **1** and the absence of pulse represents a **0**.

There are two variations in Unipolar signalling −

* Non Return to Zero NRZ
* Return to Zero RZ

### **Unipolar Non-Return to Zero NRZ**

In this type of unipolar signaling, a High in data is represented by a positive pulse called as **Mark**, which has a duration **T0** equal to the symbol bit duration. A Low in data input has no pulse.

### **Unipolar Return to Zero RZ**

In this type of unipolar signaling, a High in data, though represented by a **Mark pulse**, its duration **T0** is less than the symbol bit duration. Half of the bit duration remains high but it immediately returns to zero and shows the absence of pulse during the remaining half of the bit duration.

## **Polar Signaling**

There are two methods of Polar Signaling. They are −

* Polar NRZ
* Polar RZ

### **Polar NRZ**

In this type of Polar signaling, a High in data is represented by a positive pulse, while a Low in data is represented by a negative pulse.

### **Polar RZ**

In this type of Polar signaling, a High in data, though represented by a **Mark pulse**, its duration **T0** is less than the symbol bit duration. Half of the bit duration remains high but it immediately returns to zero and shows the absence of pulse during the remaining half of the bit duration.

However, for a Low input, a negative pulse represents the data, and the zero level remains same for the other half of the bit duration.

**Bipolar Signaling**

This is an encoding technique which has three voltage levels namely **+, -** and **0**. Such a signal is called as **duo-binary signal**.

An example of this type is **Alternate Mark Inversion**AMIAMI. For a **1**, the voltage level gets a transition from + to – or from – to +, having alternate **1s** to be of equal polarity. A **0** will have a zero voltage level.

Even in this method, we have two types.

* Bipolar NRZ
* Bipolar RZ

**Manchester Line Coding**

In telecommunication and data storage, Manchester coding (also known as Phase Encoding, or PE) is a line code in which the encoding of each data bit has at least one transition and occupies the same time. It therefore has no DC component, and is self-clocking, which means that it may be inductively or capacitively coupled, and that a clock signal can be recovered from the encoded data.

**CODE :**

clc;

pi=3.14;

x=[1,0,1,1,0,1,0,1];

nx=length(x);

sign=1;

i=1;

while i<nx+1

t=i:0.001:i+1-0.001;

if x(i)==1

unipolar\_nrz=squarewave(t\*2\*pi,100);

ami\_nrz=sign\*squarewave(t\*2\*pi,100);

unipolar\_rz=(1+squarewave(t\*2\*pi,50))/2;

polar\_rz=(1+squarewave(t\*2\*pi,50))/2;

polar\_rz=(1+squarewave(t\*2\*pi,50))/2;

ami\_rz=sign\*(1+squarewave(t\*2\*pi,50))/2;

sign=sign\*-1;

manchester\_code=squarewave(t\*2\*pi,50);

else

unipolar\_nrz=0;

ami\_nrz=0;

unipolar\_rz=0;

polar\_rz=-(1+squarewave(t\*2\*pi,50))/2;

ami\_rz=0;

manchester\_code=-squarewave(t\*2\*pi,50);

end;

subplot(4,2,1);

plot(t,unipolar\_nrz);

ylabel('unipolar\_nrz');

set(gca(),"grid",[1 1]);

a=gca();

a.data\_bounds=[1 -2;9 2]

subplot(4,2,5);

plot(t,ami\_nrz,'b');

ylabel('ami\_nrz');

set(gca(),"grid",[1 1]);

a=gca();

a.data\_bounds=[1 -2;9 2]

subplot(4,2,2);

plot(t,unipolar\_rz,'r'); ylabel('unipolar\_rz');

set(gca(),"grid",[1 1]);

a=gca();

a.data\_bounds=[1 -2;9 2]

subplot(4,2,4);

plot(t,polar\_rz);

ylabel('polar\_rz');

set(gca(),"grid",[1 1]);

a=gca();

a.data\_bounds=[1 -2;9 2]

subplot(4,2,3);

plot(t,ami\_rz,'r');

ylabel('ami\_rz');

set(gca(),"grid",[1 1]);

a=gca();

a.data\_bounds=[1 -2;9 2]

subplot(4,2,7);

plot(t,manchester\_code,'r');

ylabel('manchester\_code');

set(gca(),"grid",[1 1]);

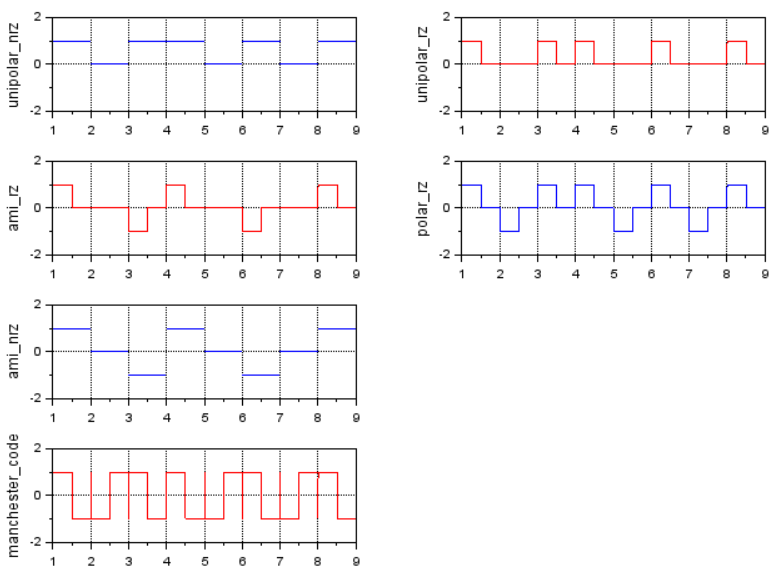
a=gca();

a.data\_bounds=[1 -2;9 2]

i=i+1;

end;

**Output 🡺**



**RESULT :**

Line coding has been studied successfully using scilab 6.1.1

**Experiment-3**

**AIM:** Write a program to study Delta Modulation and Demodulation.

**SOFTWARE USED:** Scilab 6.1.1

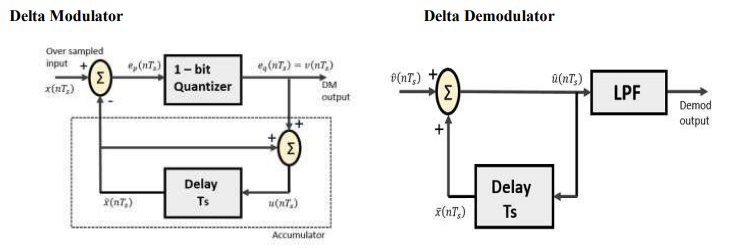
**THEORY:**

The type of modulation, where the sampling rate is much higher and in which the step size after quantization is of a smaller value **Δ**, such a modulation is termed as **delta modulation**.

Delta Modulation is a simplified form of DPCM technique, also viewed as **1-bit DPCM scheme**. As the sampling interval is reduced, the signal correlation will be higher.

**Features of Delta modulation:**

* An over-sampled input is taken to make full use of the signal correlation.
* The quantization design is simple.
* The input sequence is much higher than the Nyquist rate.
* The quality is moderate.
* The design of the modulator and the demodulator is simple.
* The stair-case approximation of output waveform



**CODE :**

pi = 3.14;

t = 0 : pi / 50 : pi;

x=sin(t);

plot (x, "r");

l = length(x);

delta = 0.2;

xn = 0;

for i = 1 : l,

if x(i) > xn(i)

d(i) = 1;

xn(i+1) = xn(i) + delta;

else

d(i) = 0;

xn(i+1) = xn(i) - delta;

end;

end;

plot2d2(xn);

for i=1:d,

if d(i ) > xn(i)

d(i) = 0;

xn(i+1) = xn(i) - delta;

else

d(i) = 1;

xn(i+1) = xn (i) + delta;

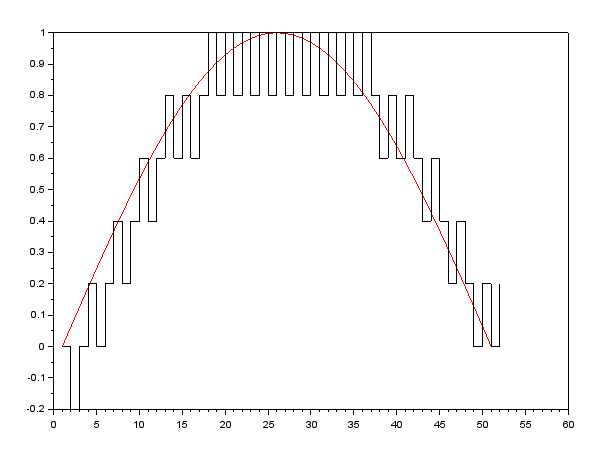
end;

end;

plot2d2(xn) ;

title("DM");

**OUTPUT :**



**RESULT :**

Delta Modulation has been studied successfully using scilab 6.1.1

**Experiment-4**

**Aim 🡺 Write a program to calculate S/N ratio and study Probability of error.**

**Software Required 🡺 Scilab**

**Theory 🡺** The signal to noise ratio formula is the [ratio](https://www.cuemath.com/commercial-math/ratio/) of the power of the signal to the power of the noise in the surrounding of the signal that is causing disturbance to the signal transmission. The signal to noise ratio formula is given as:

SNR = (Psignal)/(Pnoise) = μ / σ

where,

* Psignal is the power of the signal.
* Pnoiseis the power of the noise.
* μ is the signal mean
* σ is the standard deviation of the noise

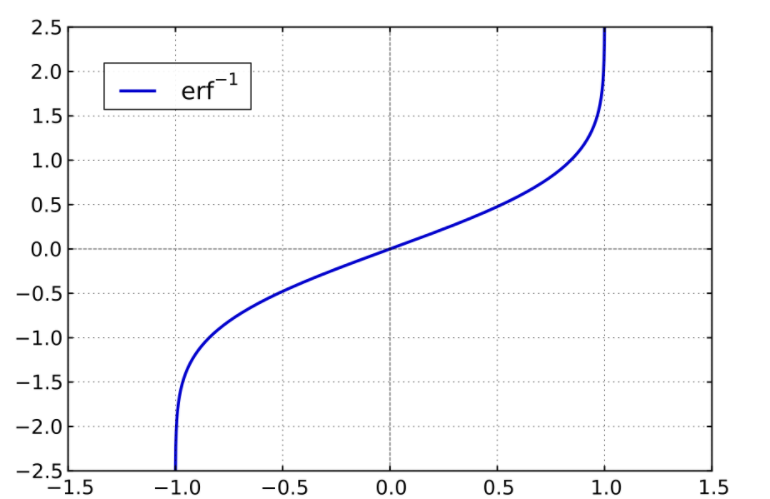
**S/N =**

**Probability of Error**

The concept of probability of error is essential in the context of digital communications and it is also sometimes known as bit error rate or BER. The BER is characterized by a bit error ratio. The bit error ratio is the ratio between the number of bit errors and the total bits transferred. The fundamentals of BER are important parameters in determining the number of bits received in a data stream constituting a communication channel. The bit errors refer to those received bits which are altered due to noise, interference, distortion, and bit synchronization. The error probability of these bits is generally evaluated using the Monte Carlo simulations. Sometimes for simpler models such as Bernoulli's distribution, analytical simulations are also preferred. Some of the other simpler models used in the information theory are the binary symmetric model and additive white gaussian noise model. The parameter, bit error is expressed as a function of Eb/No where Eb is the energy per bit and No is the noise power spectral density. For instance, in the phase shift modulation process, the bit error rate is expressed as,



where, ∈ represents the bit error rate, and erfc represents the Gauss error function. The performance of the digital communication system is determined by plotting BER curves. The BER vs received power graphs are often used in optical communication systems, where BER vs signal to ratio (SNR) curves are used for wireless communications.



**Code 🡺**

|  |  |
| --- | --- |
|  |  |

**xmax= input("Enter the value of xmax");**

**p=input("Enter the value of p");**

**n=input("Enter the value of n");**

**EBNo=10:0.1:20;**

**y=sqrt(EBNo);**

**pe=0.5\*erfc(0.5\*y);**

**SNR=3\*(2^(2\*n)\*p)/(xmax^2);**

**plot(EBNo,pe);**

**xlabel('Eb/No');**

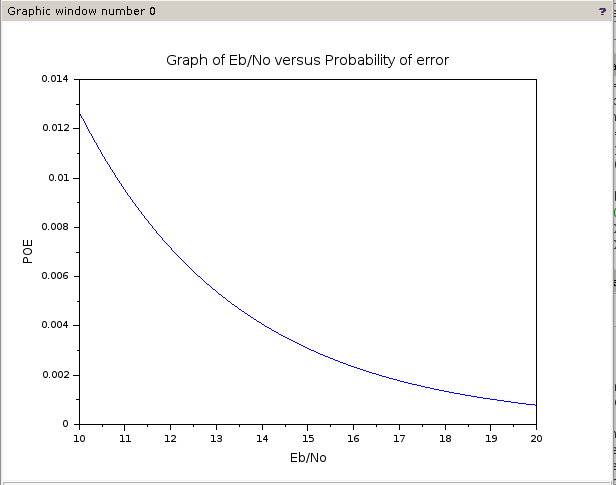
**ylabel('POE');**

**title('Graph of Eb/No versus Probability of error');**

**disp(SNR);**

**// xmax = 10, p = 100, n = 5**

**Output 🡺**



**Result 🡺** Signal to Noise Ratio (SNR) and (POE) Probability of error has been studied.

**Experiment – 5**

**Aim 🡺 Write a program to study the PDF of different random variable**

**Software Required 🡺 Scilab**

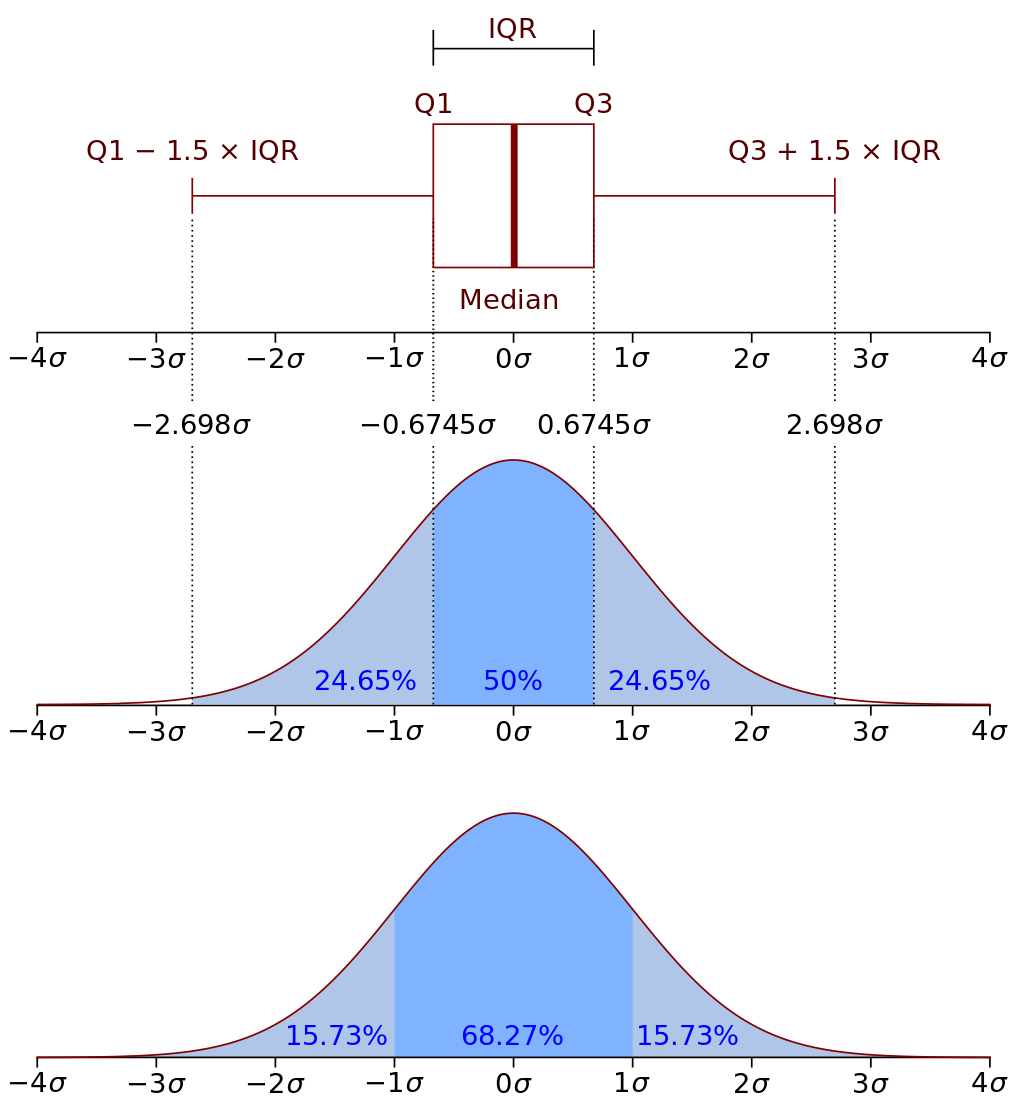
**Theory 🡺**

**PDF**

**Probability Density Function**

In probability theory, a probability density function (PDF), or density of a continuous random variable, is a function whose value at any given sample (or point) in the sample space (the set of possible values taken by the random variable) can be interpreted as providing a relative likelihood that the value of the random variable would be close to that sample Probability density is the probability per unit length, in other words, while the absolute likelihood for a continuous random variable to take on any particular value is 0 (since there is an infinite set of possible values to begin with), the value of the PDF at two different samples can be used to infer, in any particular draw of the random variable, how much more likely it is that the random variable would be close to one sample compared to the other sample.

In a more precise sense, the PDF is used to specify the probability of the random variable falling within a particular range of values, as opposed to taking on any one value. This probability is given by the integral of this variable's PDF over that range—that is, it is given by the area under the density function but above the horizontal axis and between the lowest and greatest values of the range. The probability density function is nonnegative everywhere, and the area under the entire curve is equal to 1.



**Fig: PDF of a normal distribution**

**Code 🡺**

//Experiment 5

//Aim ==> WAP to display PDF of different random variable

//13 December 2022

u=input('Mean');

v=input('Variance');

x=-10:0.1:10;

n=(1/(sqrt(2\*3.14\*v))\*exp(-(x-u)^2/(2\*v)));

figure(1);

plot(x,n);

title('Gaussian Random Variable');

a=input('Minimum Value of RV');

b=input('Maximum Value of RV');

x=a-5:0.001:b+5;

for i=1:length(x);

    if x(i)>=a&x(i)<=b

        u(i)=1/(b-a);

    else

        u(i)=0;

    end;

end;

figure(2);

plot(x,u);

title('Uniform Random Variable');

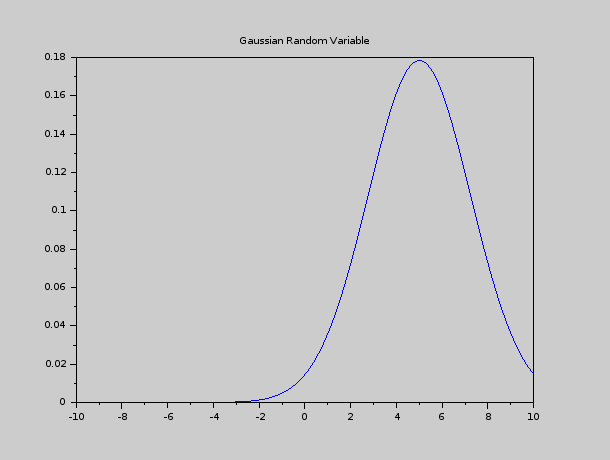
**Output 🡺**

Fig: Gaussian Random Variable

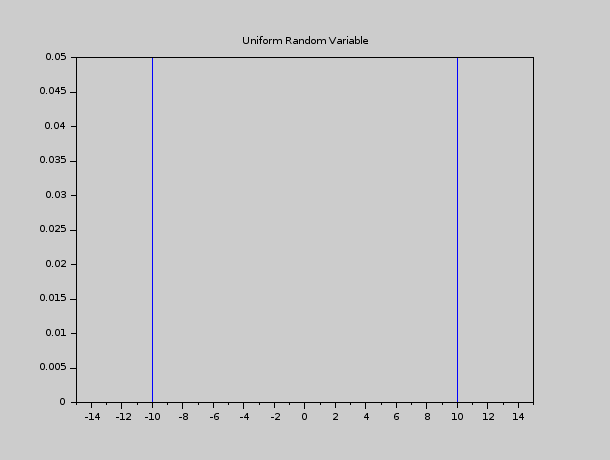


Fig: Uniform Random Variable

**Result 🡺** PDF of different random variable has been studied.