A

PROJECT BASED LAB REPORT

ON

Impact of phase on sampling and reconstruction of signals

A mini project work on Signal Analysis (15-EC-2002) submitted to
KL University under the partial fulfilment of
II /IV B. Tech

By

M. Pujitha (150030588)

MD. Sahil Afrid Farookhi (150030590)

M. Chandini Sushma (150030613)

Under the guidance of

D. BHAVANA

Assistant professor

Dept. of Computer Science Engineering
K.L. UNIVERSITY
Green fields, Vaddeswaram-522502
Guntur Dist.

2016-17

Department of Computer Science Engineering

K. L. UNIVERSITY

CERTIFICATE

This is to certify that the project entitled **"Impact of phase on sampling and reconstruction of signals"** is the bonafide work carried out by **M. Pujitha (150030588), MD. Sahil Afrid Farookhi (150030590), M. Chandini Sushma (150030613)** students of II-year B. Tech, C.S.E dept., College of Engineering, K. L. University, in the Signal Analysis Laboratory" for the academic year 2016-2017.

Signature of the Project guide   Signature of Course Coordinator

Head of the department

ACKNOWLEDGMENT

Our sincere thanks to **D. BHAVANA** in the Lab for his outstanding support throughout the project for the successful completion of the work.

My sincere thanks to **P. SASI KIRAN** Course coordinator of Signal Analysis for helping us in the completion of our project based laboratory.

We express our gratitude to **V. SRIKANTH** Head of the Department for Computer Science Engineering for providing us with adequate facilities, ways and means by which we can complete this term paper work.

We would like to place on record the deep sense of gratitude to the honourable Vice Chancellor, K. L. University for providing the necessary facilities to carry the concluded term paper work.

Last but not the least, we thank all Teaching and Non-Teaching Staff of our department and especially my classmates and my friends for their support in the completion of our project work.

Place: K. L. University

Date:

S. No	Name of the Student
1	M. Pujitha (150030588)
2	MD. Sahil Afrid Farookhi (150030590)
3	M. Chandini Sushma (150030613)

CONTENTS

ABSTRACT

Objectives:

1. Generate discrete-time sequences from analog signals for various phase angles.
2. Determine analog signals from discrete-time sequences using various interpolation filters.
3. Study the effect of phase on reconstruction signals.
4. Design a filter to remove noise

The tasks that are to be performed in this project are:

Task1: Consider an analog signal *xa*(*t*) = cos (20p*t* +f), 0 ≤*t* ≤1. Let f= 0, p / 6, p / 4, p / 3 and p / 2

Task2: This analog signal is sampled at *Ts* = 0.05 sec intervals to obtain *x*[*n*] Compute *x*[*n*] from *xa* (*t*) for all the phase values. Plot *x*[*n*] and their spectrum.

Task3: Reconstruct the analog signal *ya* (*t*) from the samples *x*[*n*] using (a) Sync (b) Cubic Spline interpolation filters. Use D*t*= 0.001 sec.

Task4: Observe the resultant construction in each case that has the correct frequency but a different amplitude. Explain these observations. Comment on the role of phase of *xa* (*t*) on the sampling and reconstruction of signals.

CHAPTER 1

1. INTRODUCTION:

A signal is defined as any physical quantity that varies with time, space, or any other independent variable or variables. Signals are classified into two types periodic signals and aperiodic signals. Periodic signals are defined as signals which repeat at time T. Aperiodic signals are defined as which don't repeat at certain intervals of time. These signals are again classified into analog and digital signals. The continuous time signal is an analog and discrete time signal is a digital signal. The signals are functions of a continuous variable, such as time or space, and usually take on values in a continuous range. Such signals may be processed directly by appropriate analog systems such as filters or frequency analyzers or

frequency multipliers for changing their characteristics or extracting some desired information. Digital signal processing provides an alternative method for processing the analog signal.

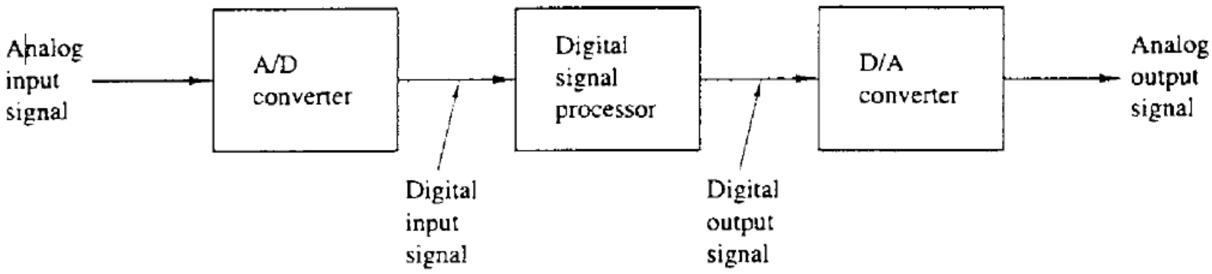

Fig.1.1 Block Diagram of digital signal processing

An analog signal is converted into a digital signal in A/D convertor by the following steps:

1. Sampling.
2. Quantising
3. coding

The sampler samples the input signal with a sampling interval T. The output signal is discrete-in-time but continuous in amplitude. The output of the sampler is applied to the quantizer. It converts the signal into discrete –time, discrete-amplitude signal. The final step is coding the coder maps each quantized sample value in digital word.

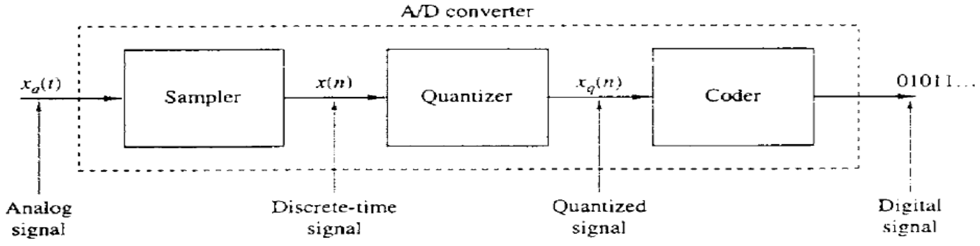

Fig.1.2 Block diagram of analog to digital conversion

1. Sampling: This is the conversion of a continuous-time signal into a discrete time signal obtained by taking “samples’" of the continuous time signal at discrete time instants.
   Thus, if xa(t) is the input to the sampler, the output is xa (nT) = x(n), w here T is called the sampling interval.
2. Quantization: This is the conversion o f a discrete-time continuous-valued signal in to a discrete-time, discrete-valued signal. The value of each signal sample is represented by a value selected from a finite set of possible values. The d difference between the un quantized sample x (n) a d the quantized output x q(n) is called the quantization error.
3. Coding: In the coding process, each discrete value x q{n) is represented by a 6-bit binary sequence.

In this we explain about the sampling and different types of sampling and how to reconstruct a signal and the impact of phase on the sampling and reconstruction of signals.

CHAPTER 2

2. SAMPLING:

In signal processing, sampling is the reduction of a continuous signal to a discrete signal. A common example is the conversion of a sound wave (a continuous signal) to a sequence of samples (a discrete-time signal). A sample refers to a value or set of values at a point in time and/or space. A sampler is a subsystem or operation that extracts samples from a continuous signal. A theoretical ideal sampler produces samples equivalent to the instantaneous value of the continuous signal at the desired points.

$$X(n) = Xa\,(nT). \quad -\infty < n < \infty$$

Where X(n) is the discrete-time signal obtained by "taking samples" of the analog signal Xa(t) every T seconds. The time interval T between successive samples is called the sampling period or sample interval and its reciprocal 1 / T = Fs is called the sampling rate.

$$T=nT=n/Fs$$

consider an analog sinusoidal signal of the form

$$Xa(t) = A \cos (2\pi Ft + \varphi)$$

When sampled periodically at a rate Fs=1 / T samples per second

$$Xa(nT) =x(n) = A \cos (2\pi FnT +\varphi) =A \cos (2\pi nF/Fs+\varphi)$$

By comparing both the Eq (1) and Eq (2) we obtain the relation between the frequency variables.

$$f=F/Fs$$

$$\text{(or) } \omega=\Omega T$$

where,

Fs=sampling frequency

F=frequency of analog

f=frequency of digital signal

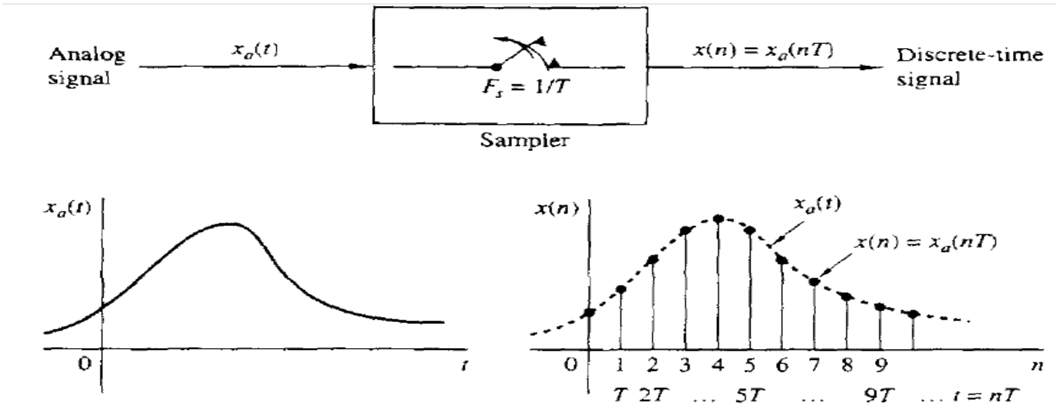

Fig 2.1 Periodic sampling of an analog signal

The range of frequency variable F or Ω for continuous sinusoidal signals are

$$-\infty < F < \infty$$

$$-\infty < \Omega < \infty$$

For a discrete sinusoidal signal,

$$-1/2 < f < 1/2$$

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

The range of the frequency of the continuous-time sinusoid when sampled at a rate Fs = 1 / T

$$-1/2T = -Fs/2 \leq F \leq Fs/2 = 1/2T$$

Equivalently

$$-\pi/T = -\pi Fs \leq \Omega \leq \pi Fs = \pi/T$$

There are two types of sampling. They are

(i) Natural sampling
(ii) Flat top sampling

Natural Sampling:

The natural sampling is one which can be represented with respect to amplitude of the analog signal.

Flat top Sampling:

The flat top sampling is the one which can be represented in only an amplitude which cannot be changed with respect to the analog signal.

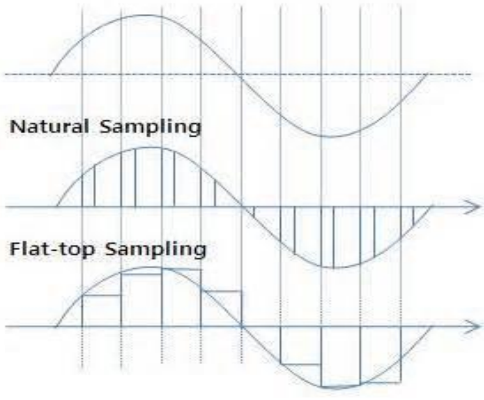

Fig.2.2 Natural Sampling and Flat-Top sampling.

Difference between them with respect to Noise:

The sample take the top signals shape (respect to amplitude of the analog signal) which mean if there is noise above signal, when it will be demodulate with LBF (low pass filter) it will cut from the original signal but in Flat top sampling **the** sample shape will be lated so if there is noise we can remove it easily and the signal we be like it transmitted without any noise.

CHAPTER 3

3. NYQUIST SAMPLING THEOREM:

A band limited signal x(t) with x(jΩ) = 0 for |Ω|=0 for |Ω|>Ωm is uniquely determined from its samples x(nT), if the sampling frequency fs≥2fmax, i.e., sampling frequency must be at least twice the highest frequency present in the signal. Where fmax is the largest frequency component in the analog signal. With the sampling rate selected in this manner, any frequency component, say |F i| < fmax, in the analog signal is mapped into a discrete-time sinusoid with a frequency.

$$Fmax=Fs/2=1/2T$$

$$\Omega max=\pi Fs=\pi/T$$

$$-1/2\leq fi=Fi/Fs\leq 1/2 \text{ or } -\pi\leq\omega i=2\pi fi\leq\pi$$

If the highest frequency contained in an analog signal Xa(t) is fmax = B and the signal is sampled at a rate Fs > 2fmax = 2B then Xa(t) can be exactly recovered from its sample values using the interpolation function.

$$g(t)=\frac{sin2\pi Bt}{2\pi Bt}$$

Thus, Xa(t) may be expressed as

$$Xa(t)=\sum_{k=-\infty}^{\infty}\left(\frac{n}{Fs}\right)Xag(t-\frac{n}{Fs})$$

When sampling of Xa(t) is performed at the minimum sampling rate Fs=2B then the reconstruction formula becomes

$$Xa(t) = \sum_{n=-\infty}^{\infty}\left(\frac{n}{2B}\right)sin2\pi B(t-\frac{n}{2B})/2\pi B(t-\frac{n}{2B})$$

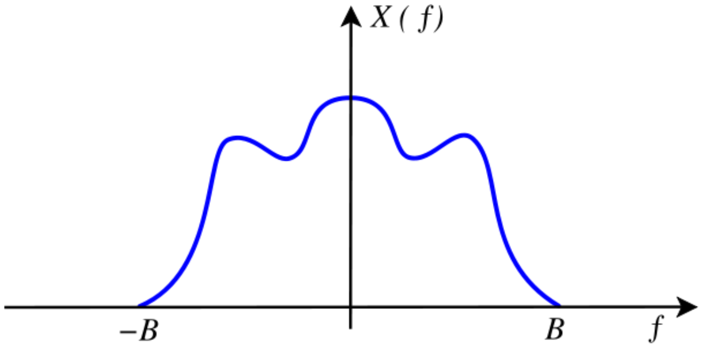

Fig.3.1 Fourier Transform of a band limited function

The sampling rate Fn==2B=2fmax is called the Nyquist rate. The Nyquist rate, named after Harry Nyquist, is twice the bandwidth of a bandlimited function or a bandlimited channel. This term means two different things under two different circumstances:

1. As a lower bound for the sample rate for alias-free signal sampling[1] (not to be confused with the Nyquist frequency, which is half the sampling rate of a discrete-time system) and
2. As an upper bound for the symbol rate across a bandwidth-limited baseband channel such as a telegraph line or passband channel such as a limited radio frequency band or a frequency division multiplex channel.

The Bandwidth is also known as the Nyquist frequency and the twice the band width is known as the Nyquist rate. The sampling frequency must be exceeded to avoid the aliasing effects.

CHAPTER 4

4. RECONSTRUCTION OF SIGNALS:

We have discussed that a band limited signal x(t) can be reconstructed from its samples if the sampling rate is Nyquist rate. This reconstruction is accomplished by passing the sampled signal through an ideal low pass filter of bandwidth *D* Hz. That sampled signal must be passed through an ideal low pass filter having bandwidth *D* Hz and gain T. This is the description for the process of reconstruction in the frequency domain to find the DTFT of the discrete-time signal.

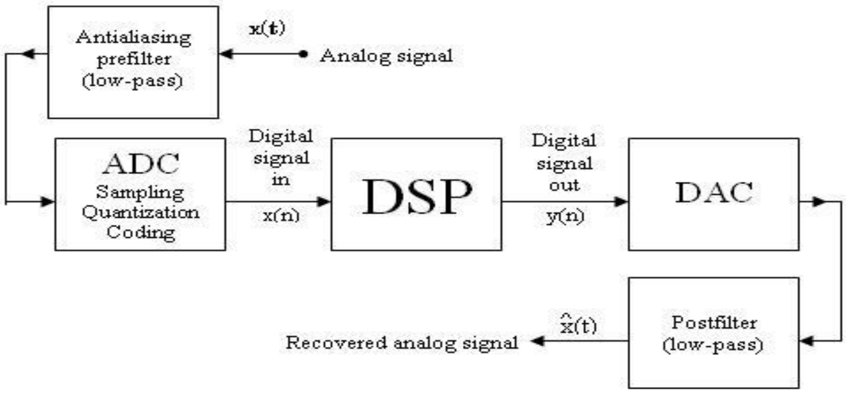

Fig.4.1 Reconstruction n the frequency domain to find DTFT

This reconstruction can be thought of as a 2-step process:

• First the samples are converted into a weighted impulse train.

$$\sum_{n=-\infty}^{\infty} x(n)\delta(t-nTs) = \cdots + x(-1)\delta(n+Ts) \,..$$

• Then the impulse train is filtered through an ideal analog low pass filter band-limited to [- Fs/2, Fs/2] band.

This two-step procedure can be described mathematically using an interpolating formula

$$Xa(t) = \sum_{n=-\infty}^{\infty} x(n)sinc|Fs(t-nTs)|$$

As per the impact of phase on reconstruction of signals,

1. Phase of a signal helps us to achieve a desired signal-noise ratio.
2. Phase plays role in the amplitude estimation stage of single-channel speech enhancement and separation.
3. Replacing the noisy signal phase with an estimated phase can lead to considerable improvement in the perceived signal quality.

There are many techniques that can be used to reconstruct a signal and the selection of the technique to be used is depends on what accuracy of reconstruction is required and how oversampled the signal is. Probably the simplest approximate reconstruction idea is to simply let the reconstruction always be the value of the most recent sample.

It is a simple technique because the samples in the form of numerical codes, can be the input signal to a Digital to Analog converter, which is clocked to produce a new output signal with every clock pulse. This technique produces a signal which has a stair step shape that follows the original signal. This type of signal reconstruction can be modelled except for quantization effects by passing the impulse sampled signal through a system called a zero-order hold. The zero-order hold causes a delay to the original signal because it is causal. Another natural reconstruction idea is to interpolate between samples with straight lines. This is obviously a better approximation of the original signal but it is a little harder to implement. This interpolation can be accomplished by following the zero orders hold by an identical zero order hold. This means that the impulse response of such a signal reconstruction filter would be the convolution of the zero order hold impulse response with itself.

4.1 Aliasing:

In reconstructing a signal from its samples, there is another practical difficulty. The sampling theorem was proved on the assumption that the signal x(t) is bandlimited. All practical signals are time limited, i.e., they are of finite duration. As a signal, cannot be time limited and bandlimited simultaneously. Thus, if a signal is time limited, it cannot be bandlimited and vice versa (but it can be simultaneously non-time limited and non-bandlimited). Clearly it can be said that all practical signals which are necessarily time limited, are non-bandlimited, they have infinite bandwidth and the spectrum $X^{'}(f)$ consists of overlapping cycles of X(f) repeating every f_s Hz (sampling frequency). Because of infinite bandwidth, the spectral overlap will always be present regardless of what ever may be the sampling rate chosen. Because of the overlapping tails$^{'}$(f) has not complete information about X(f) and it is not possible, even theoretically to recover x(t) from the sampled signal x'(t).

The loss of the tail of X(f) beyond $|f| > f_s/2$ Hz. The reappearance of this tail inverted or folded onto the spectrum. The spectra cross at frequency $f_s/2 = 1/2T$ Hz. This frequency is called the folding frequency. The spectrum folds onto itself at the folding frequency. For instance, a component of frequency $(f_s/2) + f_x$ shows up as or act like a component of lower frequency $(f_s/2)- f_x$ in the reconstructed signal. Thus, the components of frequencies above $f_s/2$ reappear as components of frequencies below $f_s/2$. This tail inversion is known as spectral folding or aliasing which is shown in Fig. 5. In this process of aliasing not only we are losing all the components of frequencies above $f_s/2$Hz, but these very components reappear as lower frequency components. This reappearance destroys the integrity of the lower frequency components.

4.2 Sampling and reconstruction in digital signal processing:

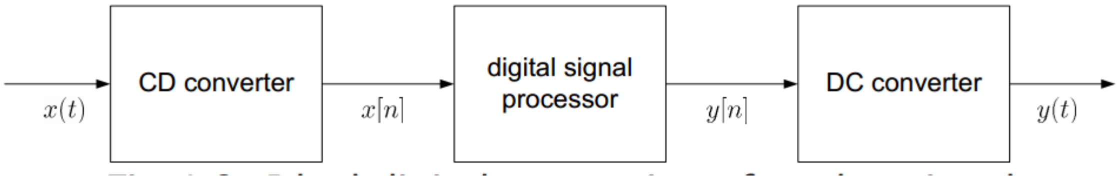

Fig.4.2 ideal digital processing of analog signal

1. CD converter produces a sequence x[n] from x(t).
2. X[n] is processed in discrete-time domain to give y(n).

$$y(t) = \sum_{k=-\infty}^{\infty} y[k]\mathrm{sinc}\left(\frac{t-kT}{T}\right)$$

3. DC converted creates y(t) from y[n].

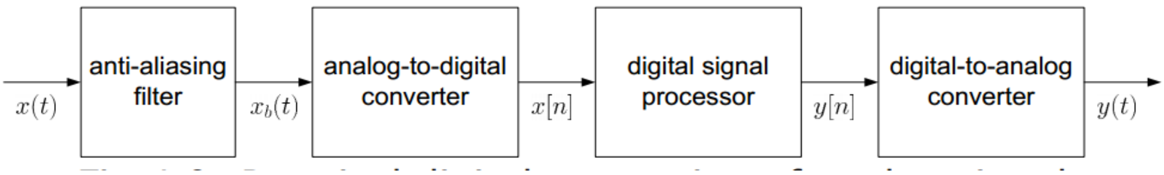

Fig.4.3 Practical digital processing of an analog signal

Anti-aliasing filter is a filter which is used before a signal sampler, to restrict the bandwidth of a signal to approximately satisfy the sampling theorem. The potential defectors are all the frequency components beyond $f_s/2$ Hz. We should have to eliminate these components from *x(t)* before sampling *x(t)*. Because of this we lose only the components beyond the folding frequency $f_s/2$ Hz. These frequency components cannot reappear to corrupt the components with frequencies below the folding frequency. This suppression of higher frequencies can be accomplished by an ideal filter of bandwidths $f/2$ Hz. This filter is called the anti-aliasing filter. The anti-aliasing operation must be performed before the signal is sampled. The anti-aliasing filter, being an ideal filter is unrealizable. In practice, we use a steep cut off filter, which leaves a sharply attenuated residual spectrum beyond the folding frequency $f_s/2$.

1. X(t) may not be precisely band limited, a low pass filter or anti-aliasing filter is needed to process x(t).
2. Ideal CD converter is approximated by AD converter
   - Not practical to generate δ(t)
   - AD converter introduces quantization error.
3. Ideal DC converter is approximated by DA converter because ideal reconstruction of is impossible
   - Not practical to perform infinite summation
   - Not practical to have future data.

CHAPTER 5

5.1 INTERPOLATION:

The process of reconstructing a continuous time signal x(t) from its samples is known as interpolation. Interpolation is often has a number of data points, obtained by the sampling or experimentation, which represent the values of a function for a limited number of values of the independent variable. It is often required to interpolate the value of that function for an intermediate value of the independent variable. This may be achieved by curve fitting or regression analysis. Another interpretation of is that it is an infinite-order interpolation. We want finite-order (and in fact low-order) interpolations.

Cubic spline Interpolation:

This approach uses spline interpolates for a smoother, but not necessarily more accurate, estimate of the analog signals between samples. Hence this interpolation does not require an analog post filter. The smoother reconstruction is obtained by using a set of piecewise continuous third-order polynomials called cubic splines. Spline interpolation is a form of interpolation where the interpolant is a special type of piecewise polynomial called a spline. Spline interpolation is often preferred over polynomial interpolation because the interpolation error can be made small even when using low degree polynomials for the spline. The most commonly used splines are cubic spline, i.e., of order 3—in particular, cubic B-spline, which is equivalent to C2 continuous composite Bezier curves. They are common, in particular, in spline interpolation simulating the function of flat splines. The term spline is adopted from the name of a flexible strip of metal commonly used by draftsmen to assist in drawing curved lines.

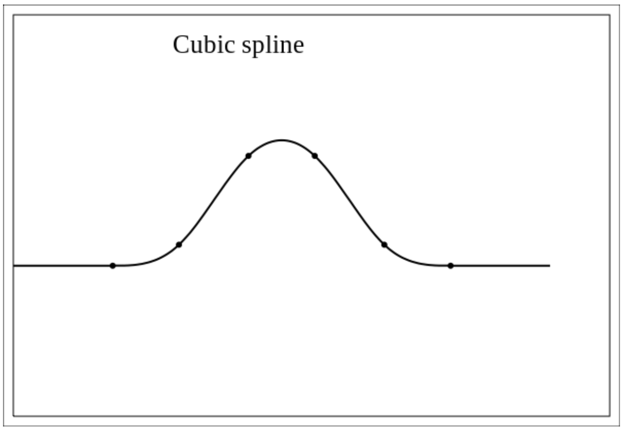

Fig 5.1 Cubic spline interpolation

A spline is a curve that's formed from a collection of polynomial segments strung end-to-end so that their junctions are smooth. If each polynomial segment has degree 3, the spline is called a cubic spline.

$$\mathrm{Xa(t)} = \alpha o(n) + \alpha 1(n)(t - nTs) + \alpha 2(n)(t - nTs)2 + \alpha 3(n)(t - nTs)3 \quad nTs \leq n < (n+1)Ts$$

where (ai (n), O $\leq$ i $\leq$ 3) are the polynomial coefficients, which are determined by using least-squares analysis on the sample values.

Sinc Interpolation:

Sinc interpolation is a method to construct a continuous-time bandlimited function from a sequence of real numbers.

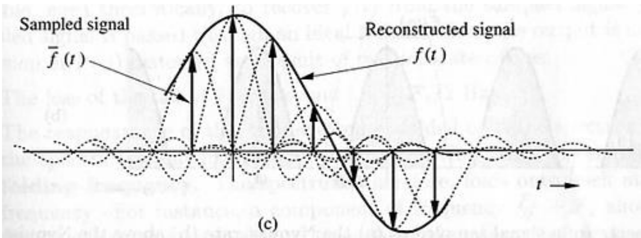

Fig 5.2 Sinc Interpolation

The function sinc(x) is defined by sinc(x)=sin(x)/x for x≠0, with sinc(0)=1. The above formula represents a linear convolution between the sequences Xn and scaled and shifted samples of the sinc function. In this Demonstration, a limited number of samples Xn are generated, and the above sum is carried out for N samples, labelled from k=0 to x=k(n-1). Due to the shifting of the sinc function by integer multiples of T, this results in x(t) having the exact value of a sample located at a multiple of T. This can be seen by observing that the absolute error is always zero at times which are integer multiples of T, in other words at the sample locations. In this implementation, the sinc function is sampled at a much higher rate than the sampling frequency used for the original function, to produce a smoother plotted result

5.2 FILTERS:

A filter is a device or process that removes from a signal some unwanted component or feature. Filters are used to remove frequencies and others to suppress interfering signals and reduce background noise. In signal processing, a filter is a device or process that removes from a signal some unwanted component or feature. Filtering is a class of signal processing, the defining feature of filters being the complete or partial suppression of some aspect of the signal. Most often, this means removing some frequencies and not others in order to suppress interfering signals and reduce background noise. However, filters do not exclusively act in the frequency domain; especially in the field of image processing many other targets for filtering exist. Correlations can be removed for certain frequency components and not for others without having to act in the frequency domain. The frequency response can be classified into a number of different band forms describing which frequency bands the filter passes (the pass band) and which it rejects:

- Low-pass filter – low frequencies are passed; high frequencies are attenuated.
- High-pass filter – high frequencies are passed; low frequencies are attenuated.
- Band-pass filter – only frequencies in a frequency band are passed.
- Band-stop filter or band-reject filter – only frequencies in a frequency band are attenuated.
- Notch filter – rejects just one specific frequency - an extreme band-stop filter.
- Comb filter – has multiple regularly spaced narrow passbands giving the band form the appearance of a comb.

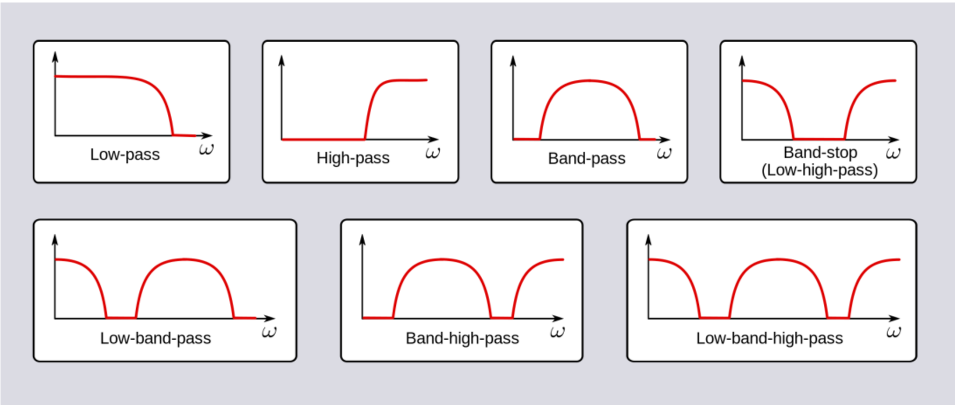

Figure 5.3 Different types of filters

- All-pass filter – all frequencies are passed, but the phase of the output is modified

CHAPTER 6

TASKS AND THEIR RESULTS:

6.1 Task1: Consider an analog signal *xa*(*t*) = cos (20p*t* +f), 0 ≤*t* ≤1. Let f= 0, p / 6, p / 4, p / 3 and p / 2

MATLAB code:

```
n=0:0.01:1;
x1=cos(20*pi*n);
x2=cos((20*pi*n)+ pi/6);
x3=cos((20*pi*n)+ pi/4);
x4=cos((20*pi*n)+ pi/3);
x5=cos((20*pi*n)+ pi/2);
subplot(5,1,1);
plot(n,x1,'r','linewidth',1.5);
title('cos(20*pi*t)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,2);
plot(n,x2,'r','LineWidth', 1.5);
title('cos(20*pi*t+pi/6)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,3);
plot(n,x3,'r','LineWidth', 1.5);
title('cos(20*pi*t+pi/4)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,4);
plot(n,x4,'r','LineWidth', 1.5);
title('cos(20*pi*t+pi/3)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,5);
```

```
plot(n,x5,'r','LineWidth', 1.5);
title('cos(20*pi*t+pi/2)');
xlabel('time');
ylabel('amplitude');
```

Result:

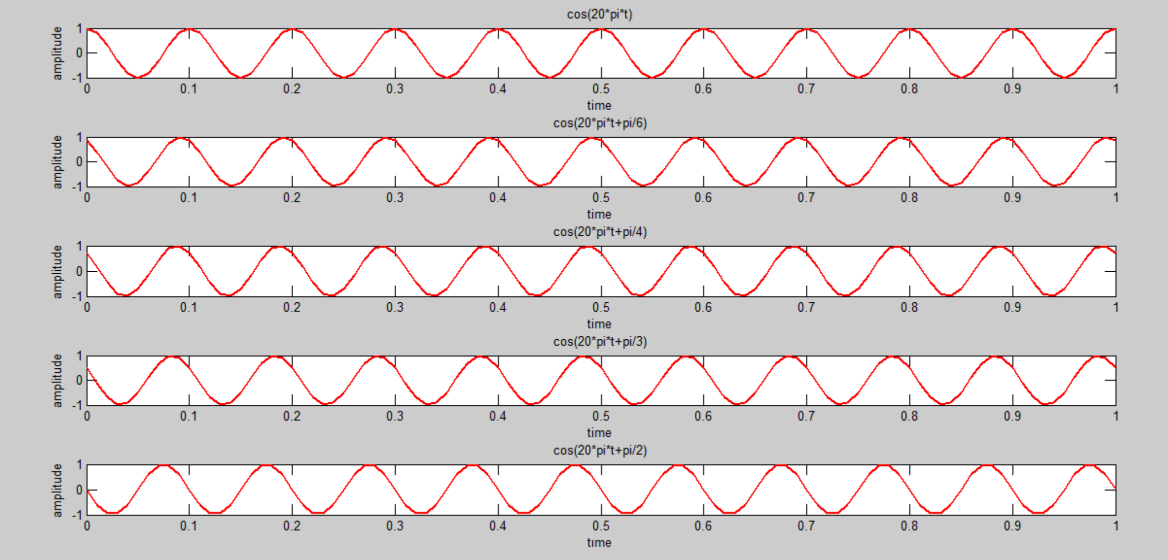

Figure6.1 Result for an analog signal

6.2 TASK 2: The analog signal is sampled at Ts = 0.05 sec intervals to obtain x[n] Compute x[n] from xa(t) from all the phase values. plt x[n] and their spectrums.

Matlab code:

```
n=0:1:20;
x1=cos(n*pi);
x2=cos(n*pi+pi/6);
x3=cos(n*pi+pi/4);
x4=cos(n*pi+pi/3);
x5=cos(n*pi+pi/2);
subplot(5,1,1);
stem(n,x1,'pb','fill','LineWidth', 1.5);
title('sampling of cos(20*pi*t)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,2);
stem(n,x2,'r', 'fill', 'LineWidth', 1.5);
title('sampling of cos(20*pi*t+pi/6)');
xlabel('time');
```

```
ylabel('amplitude');
subplot(5,1,3);
stem(n,x3,'b', 'fill', 'LineWidth', 1.5);
title('sampling of cos(20*pi*t+pi/4)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,4);
stem(n,x4,'pb', 'fill', 'LineWidth', 1.5);
title('sampling of cos(20*pi*t+pi/3)');
xlabel('time');
ylabel('amplitude');
subplot(5,1,5);
stem(n,x5,'pb', 'fill', 'LineWidth', 1.5);
title('sampling of cos(20*pi*t+pi/2)');
xlabel('time');
ylabel('amplitude');
```

Result:

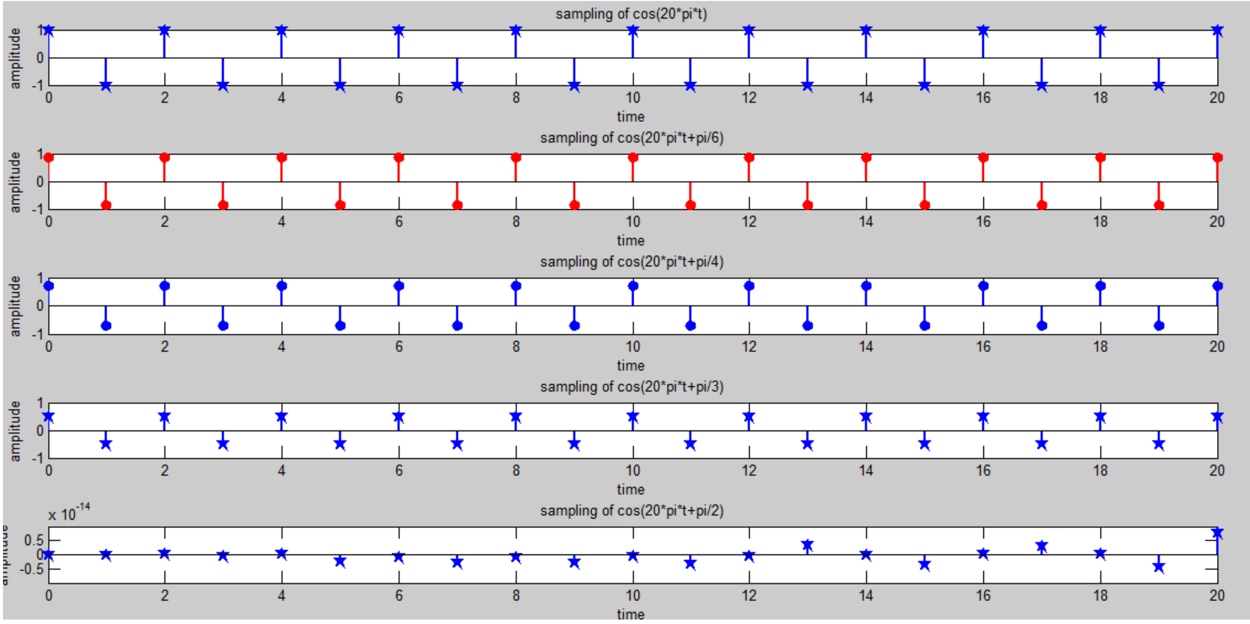

Figure 6.2 result for task2

6.3 TASK 3: Reconstruct the analog signal *ya* (*t*) from the samples *x*[*n*] using (a) Sync (b) Cubic Spline interpolation filters. Use D*t*= 0.001 sec.

Matlab code:

Sinc interpolation:

```
t=0.05;
f=1/t;
Dt=0.001;
t1=0:Dt:1;
n=0:20;
nt=n*t;
theta1=0;
x1=cos(20*pi*nt+theta1);
y1=x1*sinc(f*(ones(length(n),1)*t1-nt'*ones(1,length(t1))));
subplot(5,1,1);
plot(t1,y1,'LINEWIDTH',1.5);
hold on;
plot(nt,x1,'O');
axis([0 1 -1.2 1.2]);
xlabel('t in sec');
ylabel('amplitude');
title('sinc interpolation for theta=0');
theta2=pi/6;
x2=cos(20*pi*nt+theta2);
y2=x2*sinc(f*(ones(length(n),1)*t1-nt'*ones(1,length(t1))));
subplot(5,1,2);
plot(t1,y2,'LINEWIDTH',1.5);
hold on;
plot(nt,x2,'O');
axis([0 1 -1.2 1.2]);
xlabel('t in sec');
ylabel('amplitude');
title('sinc interpolation for theta=pi/6');
theta3=pi/4;
```

```
x3=cos(20*pi*nt+theta3);
y3=x3*sinc(f*(ones(length(n),1)*t1-nt'*ones(1,length(t1))));
subplot(5,1,3);
plot(t1,y3,'LINEWIDTH',1.5);
hold on;
plot(nt,x3,'O');
axis([0 1 -1.2 1.2]);
xlabel('t in sec');
ylabel('amplitude');
title('sinc interpolation for theta=pi/4');
theta4=pi/3;
x4=cos(20*pi*nt+theta4);
y4=x4*sinc(f*(ones(length(n),1)*t1-nt'*ones(1,length(t1))));
subplot(5,1,4);
plot(t1,y4,'LINEWIDTH',1.5);
hold on;
plot(nt,x4,'O');
axis([0 1 -1.2 1.2]);
xlabel('t in sec');
ylabel('amplitude');
title('sinc interpolation for theta=pi/3');
theta5=pi/2;
x5=cos(20*pi*nt+theta5);
y5=x5*sinc(f*(ones(length(n),1)*t1-nt'*ones(1,length(t1))));
subplot(5,1,5);
plot(t1,y5,'LINEWIDTH',1.5);
hold on;
plot(nt,x5,'O');
axis([0 1 -1.2 1.2]);
xlabel('t in sec');
ylabel('amplitude');
title('sinc interpolation for theta=pi/2');
```

Spline interpolation:

```
t=0.05;
Dt=0.001;
t1=0:Dt:1;
n=0:20;
nt=n*t;
theta1=0;
x1=cos(20*pi*nt+theta1);
y=spline(nt,x1,t1);
subplot(5,1,1);
plot(t1,y,'LINEWIDTH',1.5);
axis([0 1 -1.2 1.2]);
hold on;
plot(nt,x1,'O');
xlabel('t in sec');
title('spline interpolation for theta=0');
ylabel('amplitude');
theta2=pi/6;
x2=cos(20*pi*nt+theta2);
y1=spline(nt,x2,t1);
subplot(5,1,2);
plot(t1,y1,'LINEWIDTH',1.5);
axis([0 1 -1.2 1.2]);
hold on;
plot(nt,x2,'O');
xlabel('t in sec');
title('spline interpolation for theta=pi/6');
ylabel('amplitude');
theta3=pi/4;
x3=cos(20*pi*nt+theta3);
y2=spline(nt,x3,t1);
subplot(5,1,3);
plot(t1,y2,'LINEWIDTH',1.5);
axis([0 1 -1.2 1.2]);
```

```
hold on;
plot(nt,x3,'O');
xlabel('t in sec');
title('spline interpolation for theta=pi/4');
ylabel('amplitude');
theta4=pi/3;
x4=cos(20*pi*nt+theta4);
y3=spline(nt,x4,t1);
subplot(5,1,4);
plot(t1,y3,'linewidth',1.5);
axis([0 1 -1.2 1.2]);
hold on;
plot(nt,x4,'O');
xlabel('t in sec');
title('spline interpolation for theta=pi/3');
ylabel('amplitude');
theta5=pi/2;
x5=cos(20*pi*nt+theta5);
y4=spline(nt,x5,t1);
subplot(5,1,5);
plot(t1,y4,'LINEWIDTH',1.5);
axis([0 1 -1.2 1.2]);
hold on;
plot(nt,x5,'O');
xlabel('t in sec');
title('spline interpolation for theta=pi/2');
ylabel('amplitude');
```

Result:

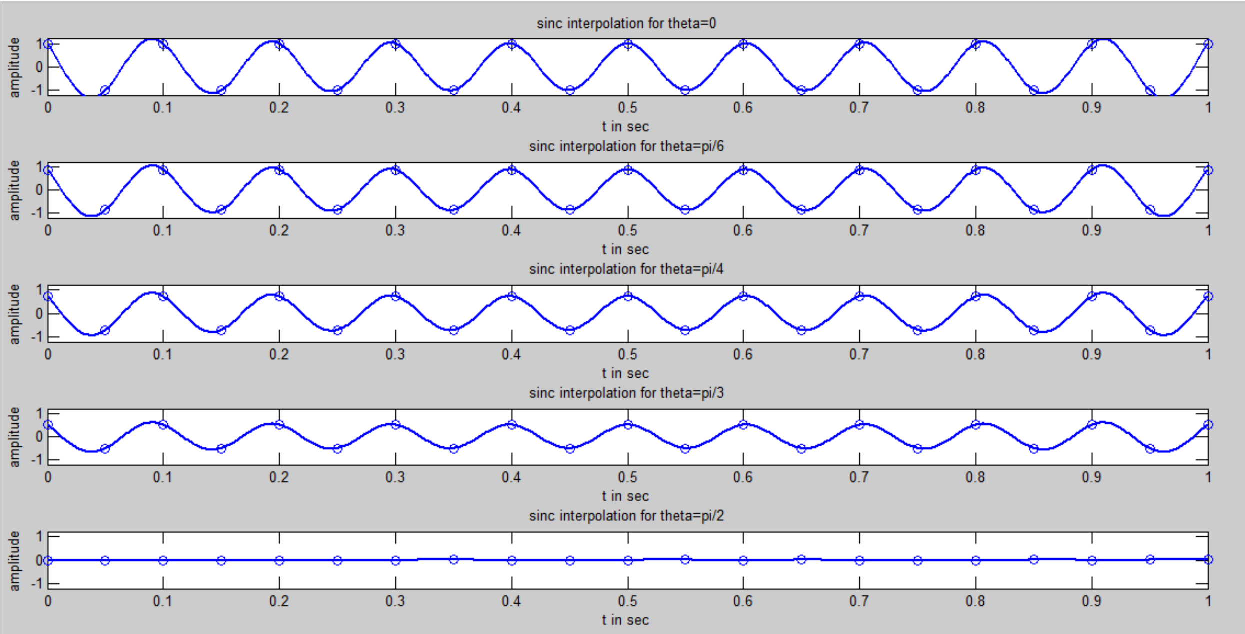

Figure 6.31 sinc interpolation

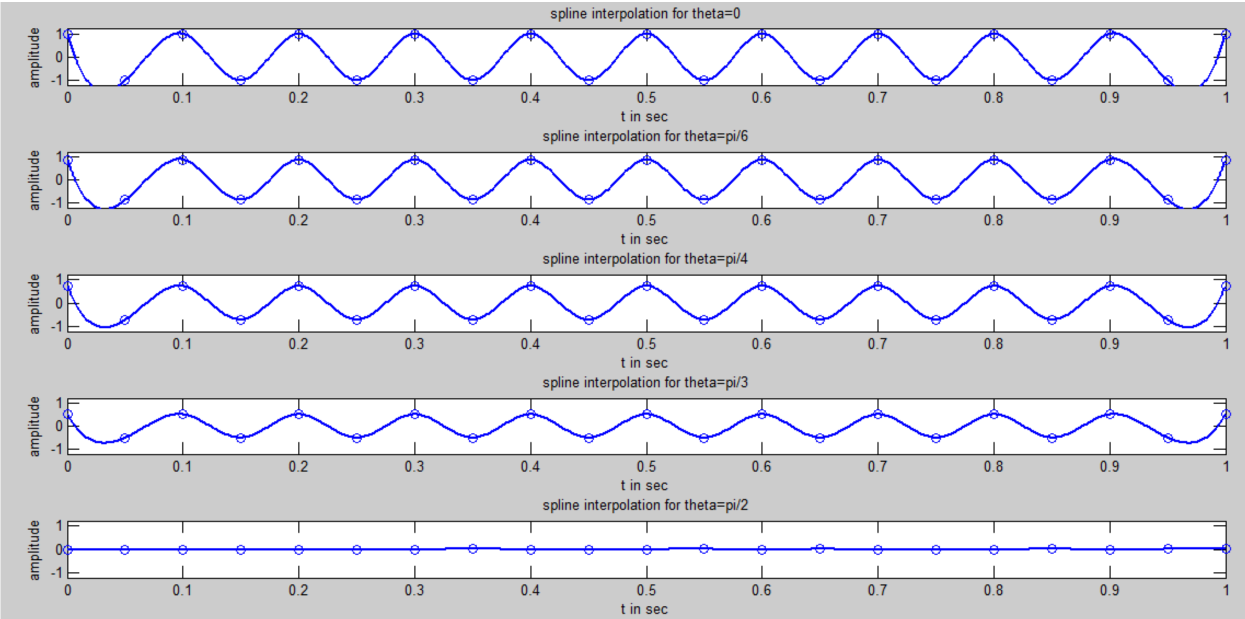

Figure 6.3 Spline interpolation

6.4 TASK 4: Observe the resultant construction in each case that has the correct frequency but a different amplitude. Explain these observations. Comment on the role of phase of *xa* (*t*) on the sampling and reconstruction of signals.

Result:

When a sinusoidal signal is sampled at f=2samples per cycle as in this case, then the resulting amplitude x (n) has the amplitude that depends on the phase of the signal. This amplitude is given by cos (theta). Thus, the amplitude of the reconstructed signal y (t) is also equal to cos (theta).

CONCLUSION:

By doing this project we have understood about the sampling and the impact of the phase on the sampling and reconstruction of signals. We have taken an analog signal and then sampled it at certain phase and reconstructed the sampled signal at a time and analysed the spectrum signal. We have reconstructed the signal by using various interpolation methods. We have understood of sampling, reconstruction of signal, interpolation, filters and and the impact of phase on sampling and reconstruction of signals.

REFERENCES:

[I]Solution-Manual-Digital-Signal-Processing-Using-Matlab-Proakis-2-edition.

[2] Digital signal processing using matlab by proakis and vinay. k 3rd edition.

[3] Digital signal processing by proakis 3rd edition.

[4] F. Kaiser, "Nonrecursive digital filter design using IO- sinh window function" in Proc. IEEE Int. Symp. Circuits and Systems (ISCAS74), San Francisco, Calif, USA, 1974, pp.20-23