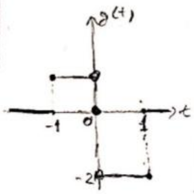
Emre Can Şen- 21902516

Section-2

Lab-05

Analytical Part

Part-1

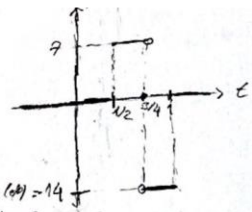


$f(t) = 7g(4t-3)$

1- $g(t-3) = f_1(t)$ → right shift by 3

2- $f_1(4t) = f_2(t) \Rightarrow$ → scale x-axis

3- $7f_2(t) = f(t)$

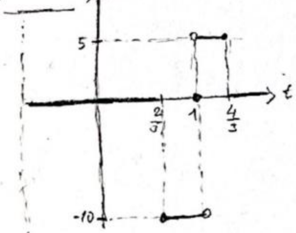


$h(t) = 5g(-3(t-1))$

1- $g(t+3) = h_1(t)$ (shift 3 by left)

2- $h_2(-3t) = h_3(t)$ (scale & mirror)

3- $5h_3(t) = h(t)$



$-1 \le \frac{t}{T_s} \le 1$

1-) $g(t) \xrightarrow{\mathcal{F}} G(\omega) = \int_{-1}^{0} e^{-j\omega t}dt - 2\int_{0}^{1} e^{-j\omega t}dt$

$= \frac{e^{j\omega}-1}{j\omega} + \frac{2e^{-j\omega}-2}{j\omega} = \frac{2e^{-j\omega}+e^{j\omega}-3}{j\omega}$

$= \frac{3\cos(\omega) - j\sin(\omega) - 3}{j\omega} =$

$\Rightarrow \frac{j3(1-\cos(\omega))}{\omega} - \frac{\sin(\omega)}{\omega} = \frac{3j\sin(\omega)}{\omega} - \frac{\sin(\omega)}{\omega}$ • Or it can be interpreted as superposition of two square wave fourier t.: sinc functions. Since sinc function is not periodic, there will be aliasing. Same case applies to other signals too.

• $\tilde{x}[n] = x(nT_s) \Rightarrow \tilde{x}(t) * p(t) = \int_{-\infty}^{\infty} \tilde{x}(\tau)\,p(t-\tau)\,d\tau$, $\tau = n'T_s \Rightarrow \int_{-\infty}^{\infty} \tilde{x}(n'T_s)\,p(t-n'T_s)\,d(n'T_s) = \sum_{n'=-\infty}^{\infty} x[n']\,p(t-n'T_s) = x_R(t)$

$\sum_{n=-\infty}^{\infty} \tilde{x}[n] = \int_{-\infty}^{\infty} x(nT_s)\,d(nT_s)$ (eq.1)

• $x_R(nT_s) = \sum_{n'=-\infty}^{\infty} \tilde{x}[n']\,p(nT_s - n'T_s) = \sum_{n'=-\infty}^{\infty} \tilde{x}[n']\,p(T_s\underbrace{(n-n')}_{k})$, $k=0 \Rightarrow n=n'$, $k\neq 0 \Rightarrow n\neq n'$ → in which case all $p(T_s(n-n'))$ are 0.

$\xrightarrow{n'=n} x_R(nT_s) = \tilde{x}[n]$

a-) $p_Z(0) = 1$, $p_L(0) = 1$, $p_I(0) = 1$ b-) $p_Z(kT_s) = rect(k) = \begin{cases} 1, & -\frac{1}{2} < k < \frac{1}{2} \\ 0, & \text{o.w.} \end{cases}$

$p_L(kT_s) = tri(\frac{k}{2}) = \begin{cases} 1-|k|, & -1 < k \le 1 \\ 0, & \text{o.w.} \end{cases}$, $p_I(kT_s) = sinc(k) = \begin{cases} 1, & k=0 \\ \frac{\sin(\pi k)}{\pi k}, & \text{o.w.} \end{cases}$

c-) No, interpolations will be different from one another.

Figure 1- Analytical Part 1

Part-5

* In the previous parts T_s denotes sampling frequency but here it seems to be sampling interval. I am not sure which one we are increasing, but if the sampling freq. is being increased; then since there is more data sampled, the quality will increase. If it is sampling period, inverse will be true.

* First type of interpolation is the most successfull, since $g(t)$ is also a combination of square wave signals. Type 2 & 3 are way off compared to type 1. It was expected that most similar signal, in this case type 1, would give the most accurate reconstruction.

Part - 6

a-) Since this is a periodic signal, type 0 & 1 will work better than type 3. For, type 0 & 1 type 0 works best because it almost looks like one of the periods. If correct intervals are taken, the period can be obtained and then the whole signal.

b-) Sampling rate dropped considerably & interpolated signals look more alike because of lack of detail. It is really hard to tell which one is the ideal interpolator.

c-) Now signal is aligned such that the negative parts are more frequent in the sampled version between about -3 & 2.5. Then positive parts are more frequent. Because type 2 incorporates more points than the others, the change to positive can be observed. Also type 0 and type 2's common parts are very alike. But since type 2 has additional information it is more accurate. Also type 1 will be the worst, because it is almost the opposite of others.

d-) Again, common parts of type 0 & 2 are very alike but type 2 has more information about the rest of the signal. But since sampling rate has not dropped as much as part - c, the additional data will be more akin to noise.

* Between $0.01 < T_s < 0.1$ the interpolated signal quality decreases but the interpolated signal is close to the original. The interpolations just get sharper, more corners rather than smooth edges. But after that, $w_{max} = 10\pi$, $f_{max} = 5Hz$, $T_{max} = 0.2 \rightarrow$ hence nyquist rate is reached after $T_s = 0.1$. When sampling period gets larger than 0.1s, there will be aliasing & data loss. So the quality of interpolations drop significantly after that.

Figure 2- Analytical Part 2

<u>MATLAB Plots</u>

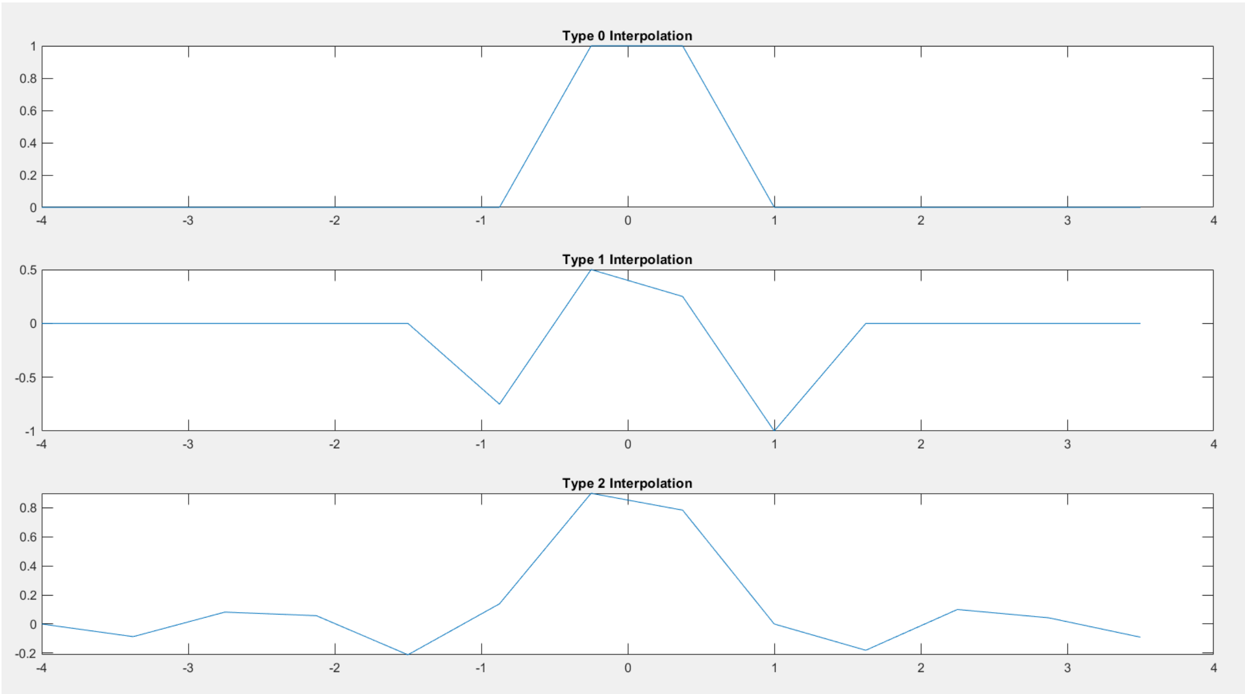


Figure 3- Ts=8/5 Interpolation Graphs

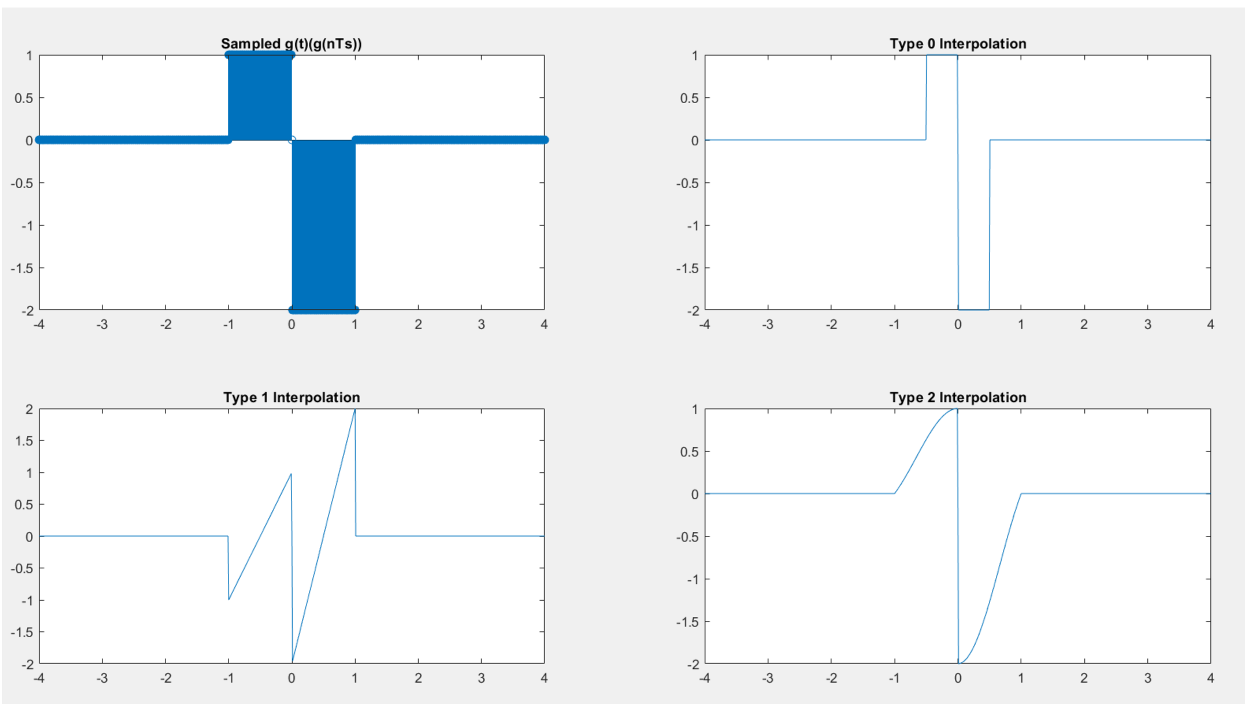


Figure 4- Part 5 Graphs

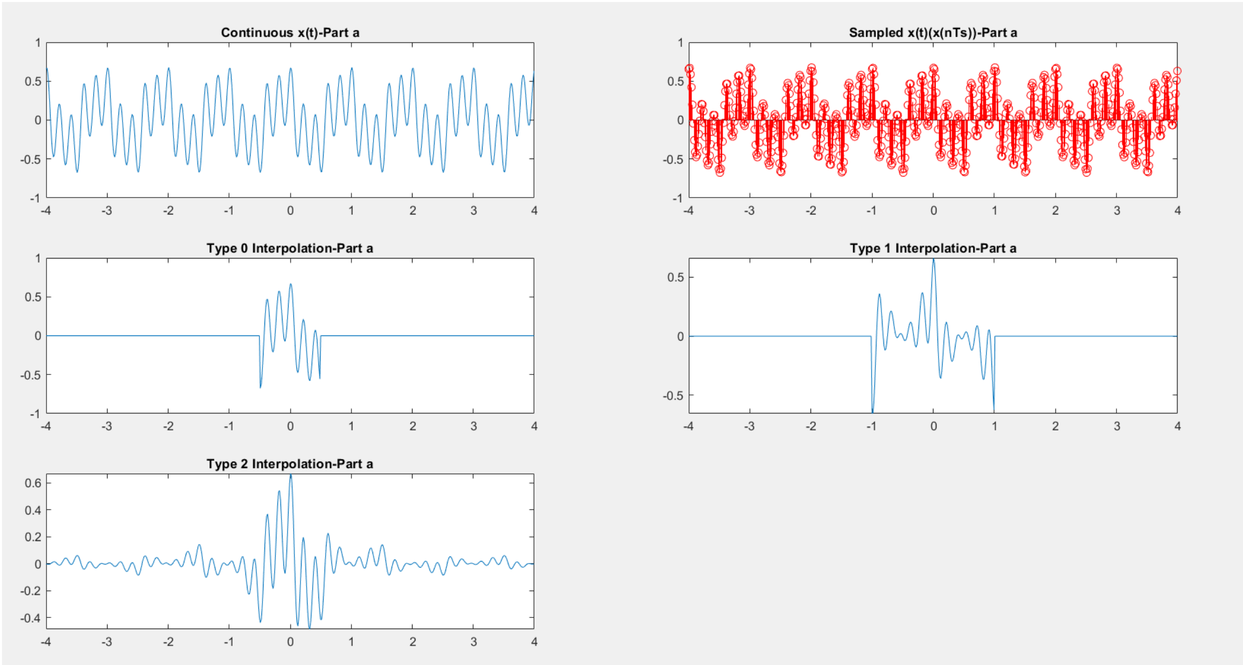


Figure 5- Part 6-A

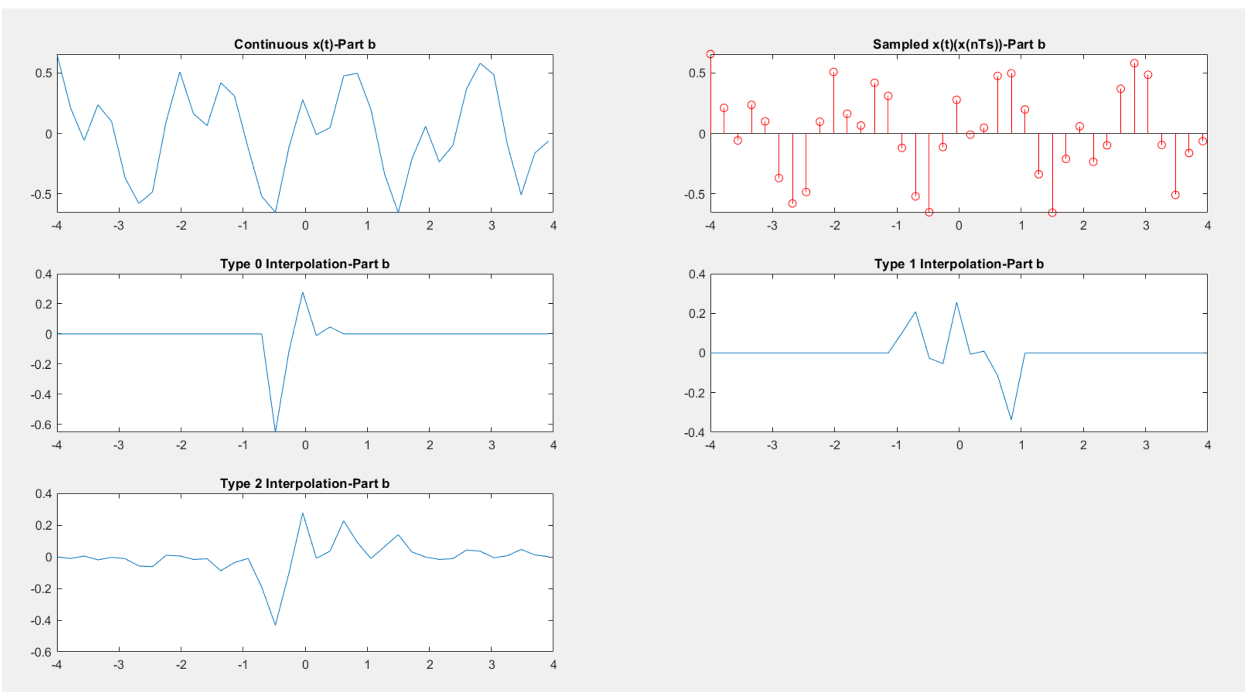


Figure 6- Part 6-B

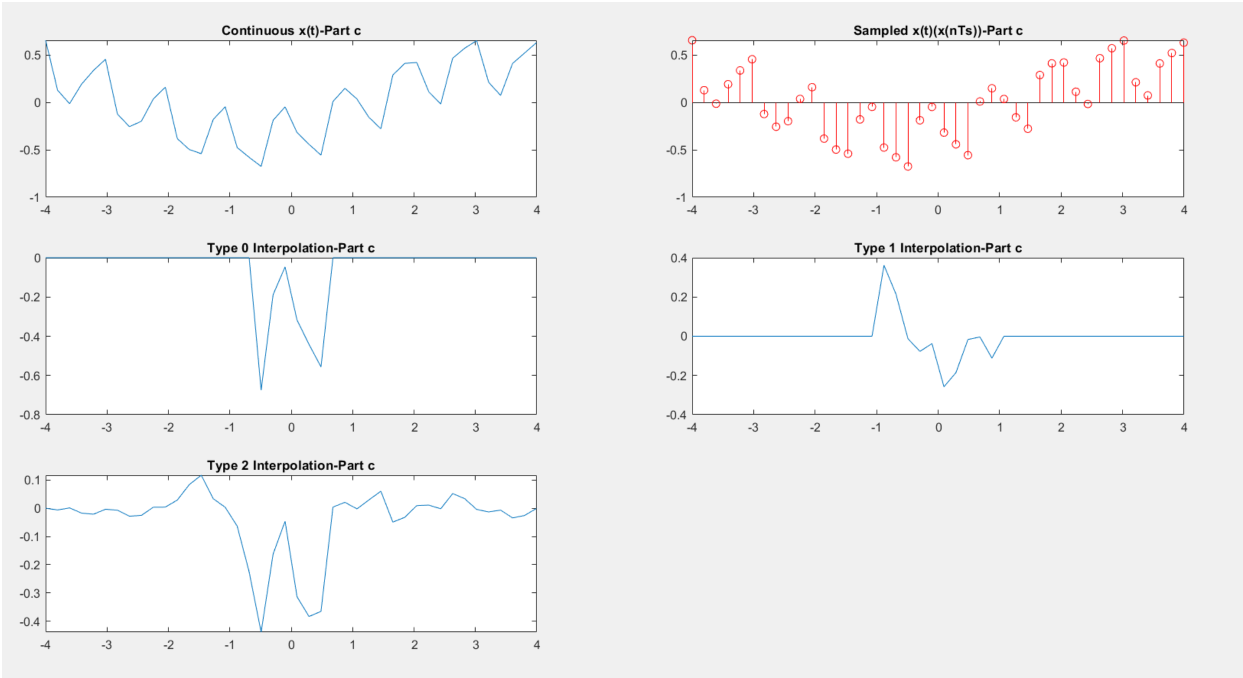


Figure 7- Part 6-C

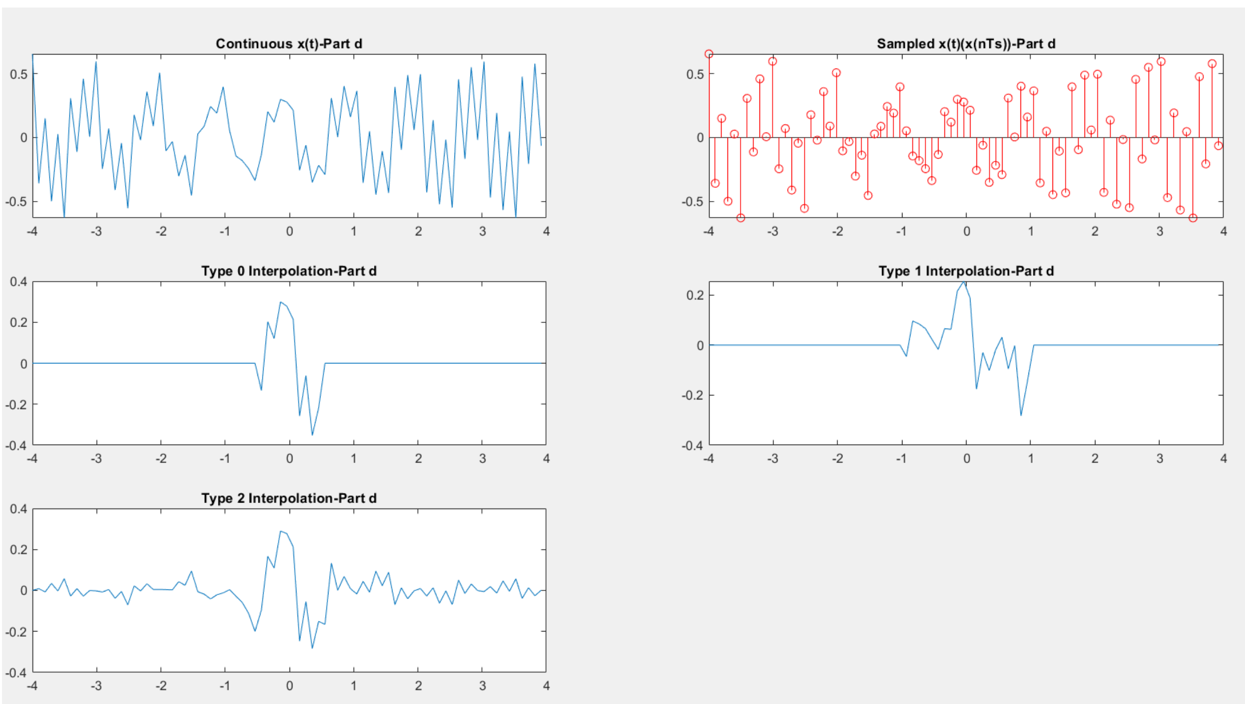


Figure 8- Part 6-D

MATLAB Code

```
p=generateInterp(0,8/5,8);
dur=8;
Ts=8/5;
type=0;
T0=-dur/2:1/Ts:dur/2;
figure(1)

subplot(3,1,1)
plot(T0,p)
title('Type 0 Interpolation')

subplot(3,1,2)
p=generateInterp(1,8/5,8);
plot(T0,p)
title('Type 1 Interpolation')

subplot(3,1,3)
p=generateInterp(2,8/5,8);
plot(T0,p)
title('Type 2 Interpolation')


r = randi([3 8],1);
Ts1_T=1/(15*r);
Ts1=1/Ts1_T;
type=0;
N=-4:1/Ts1:4;
g_x=zeros(1,length(N));
T_1=(-1<=N & N<0);
T_2=(0<N & N<=1);
g_x(T_1)=g_x(T_1)+1;
g_x(T_2)=g_x(T_2)-2;
dur0=8;

figure(2)
subplot(2,2,1)
stem(N,g_x)
title('Sampled g(t)(g(nTs))')

xR=DtoA(type,Ts1,dur0,g_x);
```

```
subplot(2,2,2)
plot(N,xR)
title('Type 0 Interpolation')

subplot(2,2,3)
xR=DtoA(1,Ts1,dur0,g_x);
plot(N,xR)
title('Type 1 Interpolation')

subplot(2,2,4)
xR=DtoA(2,Ts1,dur0,g_x);
plot(N,xR)
title('Type 2 Interpolation')

D=2;
TsX_T=(0.005)*(D+1);
TsX=1/TsX_T;
N0=-4:1/TsX:4;
x_t=zeros(1,length(N0));
T_3=((-4)<=N0 & N0<=4);
x_t(T_3)=x_t(T_3)+(0.3)*cos(2*pi*N0(T_3)+pi/4)+(0.1)*cos(6*p
i*N0(T_3)+pi/8)+(0.4)*cos(10*pi*N0(T_3)-1/2);

figure(3)
subplot(3,2,1)
plot(N0,x_t)
title('Continuous x(t)-Part a')

subplot(3,2,2)
stem(N0,x_t,'r')
title('Sampled x(t)(x(nTs))-Part a')

subplot(3,2,3)
xR=DtoA(0,TsX,dur0,x_t);
plot(N0,xR)
title('Type 0 Interpolation-Part a')

subplot(3,2,4)
xR=DtoA(1,TsX,dur0,x_t);
plot(N0,xR)
title('Type 1 Interpolation-Part a')

subplot(3,2,5)
xR=DtoA(2,TsX,dur0,x_t);
```

```
plot(N0,xR)
title('Type 2 Interpolation-Part a')

TsX_T=0.2+(0.01)*D;
TsX=1/TsX_T;
N0=-4:1/TsX:4;
x_t=zeros(1,length(N0));
T_3=((-4)<=N0 & N0<=4);
x_t(T_3)=x_t(T_3)+(0.3)*cos(2*pi*N0(T_3)+pi/4)+(0.1)*cos(6*p
i*N0(T_3)+pi/8)+(0.4)*cos(10*pi*N0(T_3)-1/2);

figure(4)
subplot(3,2,1)
plot(N0,x_t)
title('Continuous x(t)-Part b')

subplot(3,2,2)
stem(N0,x_t,'r')
title('Sampled x(t)(x(nTs))-Part b')

subplot(3,2,3)
xR=DtoA(0,TsX,dur0,x_t);
plot(N0,xR)
title('Type 0 Interpolation-Part b')

subplot(3,2,4)
xR=DtoA(1,TsX,dur0,x_t);
plot(N0,xR)
title('Type 1 Interpolation-Part b')

subplot(3,2,5)
xR=DtoA(2,TsX,dur0,x_t);
plot(N0,xR)
title('Type 2 Interpolation-Part b')


TsX_T=0.18+(0.005)*(D+1);
TsX=1/TsX_T;
N0=-4:1/TsX:4;
x_t=zeros(1,length(N0));
T_3=((-4)<=N0 & N0<=4);
x_t(T_3)=x_t(T_3)+(0.3)*cos(2*pi*N0(T_3)+pi/4)+(0.1)*cos(6*p
i*N0(T_3)+pi/8)+(0.4)*cos(10*pi*N0(T_3)-1/2);
```

```
figure(5)
subplot(3,2,1)
plot(N0,x_t)
title('Continuous x(t)-Part c')

subplot(3,2,2)
stem(N0,x_t,'r')
title('Sampled x(t)(x(nTs))-Part c')

subplot(3,2,3)
xR=DtoA(0,TsX,dur0,x_t);
plot(N0,xR)
title('Type 0 Interpolation-Part c')

subplot(3,2,4)
xR=DtoA(1,TsX,dur0,x_t);
plot(N0,xR)
title('Type 1 Interpolation-Part c')

subplot(3,2,5)
xR=DtoA(2,TsX,dur0,x_t);
plot(N0,xR)
title('Type 2 Interpolation-Part c')


TsX_T=0.099;
TsX=1/TsX_T;
N0=-4:1/TsX:4;
x_t=zeros(1,length(N0));
T_3=((-4)<=N0 & N0<=4);
x_t(T_3)=x_t(T_3)+(0.3)*cos(2*pi*N0(T_3)+pi/4)+(0.1)*cos(6*p
i*N0(T_3)+pi/8)+(0.4)*cos(10*pi*N0(T_3)-1/2);

figure(6)
subplot(3,2,1)
plot(N0,x_t)
title('Continuous x(t)-Part d')

subplot(3,2,2)
stem(N0,x_t,'r')
title('Sampled x(t)(x(nTs))-Part d')

subplot(3,2,3)
xR=DtoA(0,TsX,dur0,x_t);
```

```
plot(N0,xR)
title('Type 0 Interpolation-Part d')

subplot(3,2,4)
xR=DtoA(1,TsX,dur0,x_t);
plot(N0,xR)
title('Type 1 Interpolation-Part d')

subplot(3,2,5)
xR=DtoA(2,TsX,dur0,x_t);
plot(N0,xR)
title('Type 2 Interpolation-Part d')



function p=generateInterp(type,Ts,dur)
    T0=-dur/2:1/Ts:dur/2;
    p=zeros(1,length(T0));
    Tz=((-1/2)<=T0 & T0<=(1/2));
    Tl=(-1<=T0 & T0<=1);
    Ti=(T0~=0);
    Ti1=(T0==0);
    if type==0
        p(Tz)=p(Tz)+1;
    elseif type==1
        p(Tl)=p(Tl)+1-2.*abs(T0(Tl));
    elseif type==2
        p(Ti)=p(Ti)+sin(pi*T0(Ti))./(pi*T0(Ti));
        p(Ti1)=p(Ti1)+1;
    end
end

function xR=DtoA(type,Ts,dur,Xn)
    p_t=generateInterp(type,Ts,dur);
    xR=zeros(1,length(p_t));
    xR=xR+Xn.*p_t;
end
```