BILKENT UNIVERSITY

COMPUTER ENGINEERING

EEE391-BASICS OF SIGNALS AND SYSTEMS

MATLAB ASSIGNMENT 1

Sena Korkut

21703303

24/11/2020

Introduction

In this report, several different interpolation pulses will be analyzed and discussed on two different discrete time sequences.

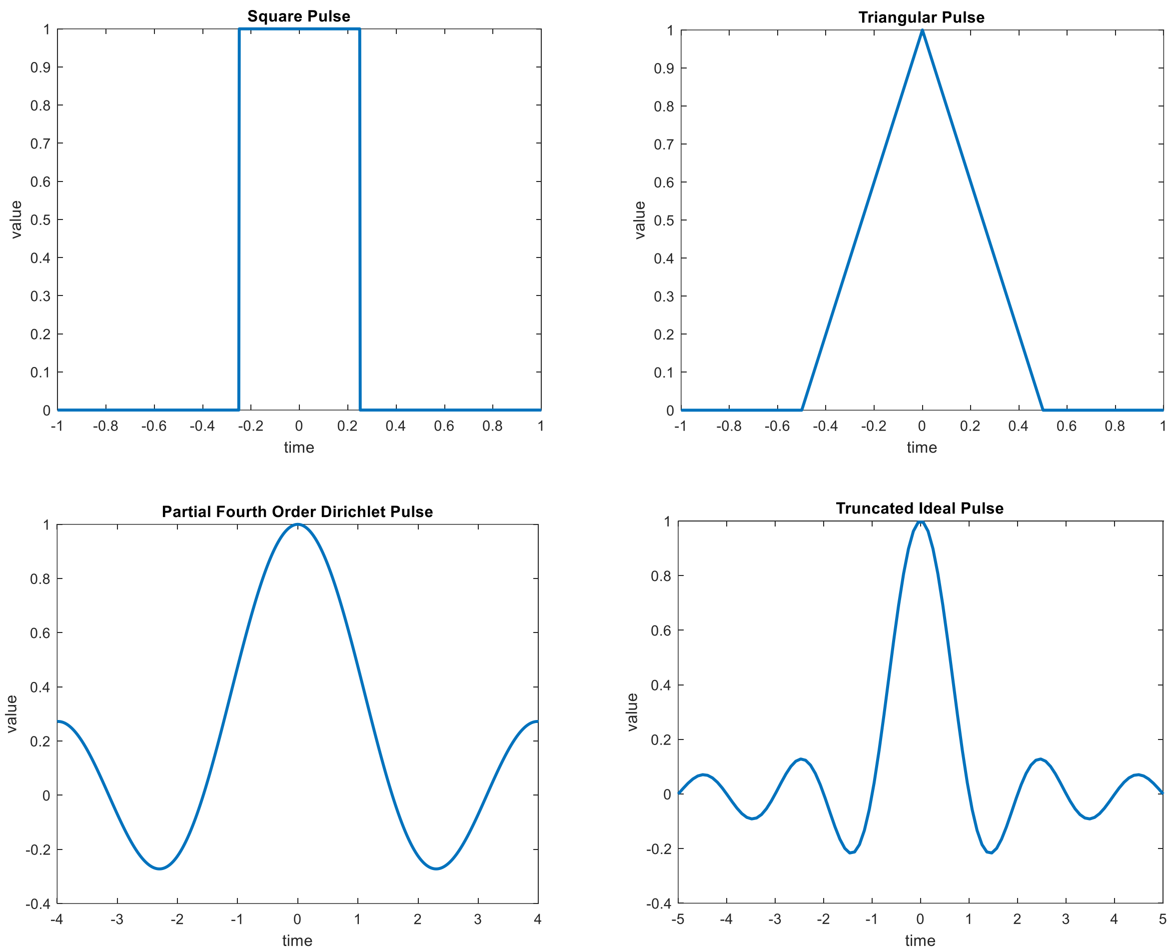

Figure 1: Different Pulse Types for Interpolation

For this assignment, signal reconstruction will be done by using following equation:

$$y(\mathrm{t}) = \sum_{n=-\infty}^{\infty} \left(y[n]p(t - n\mathrm{T_s})\right)$$

Four different pulse types in *Figure 1* will be used in experiments. Followings are mathematical function that represent pulse signals above.

Square Pulse:

$$p(\mathrm{t}) = \begin{cases} 1, & {}^{-\mathrm{T_s}}/_{2} \leq \mathrm{t} \leq {}^{\mathrm{T_s}}/_{2} \\ 0, & \text{otherwise} \end{cases}$$

Triangular Pulse:

$$p(\mathrm{t}) = \begin{cases} 1- {}^{|\mathrm{t}|}/_{\mathrm{T_s}}, & -\mathrm{T_s} \leq \mathrm{t} \leq \mathrm{T_s} \\ 0, & \text{otherwise} \end{cases}$$

Partial Fourth Order(n=4) Dirichlet Pulse:

$$p(\mathrm{t}) = \begin{cases} \dfrac{1\text{-}\sin\frac{\pi \mathrm{t}}{\mathrm{T_s}}}{\frac{\pi \mathrm{t}}{\mathrm{T}}}, & -2\mathrm{T_s} \leq \mathrm{t} \leq 2\mathrm{T_s} \\ 0, & \text{otherwise} \end{cases}$$

Truncated Ideal Pulse:

$$p(\mathrm{t}) = \begin{cases} \dfrac{1\text{-}\sin\frac{\pi \mathrm{t}}{\mathrm{T_s}}}{\frac{\pi \mathrm{t}}{\mathrm{T}}}, & -3\mathrm{T_s} \leq \mathrm{t} \leq 3\mathrm{T_s} \\ 0, & \text{otherwise} \end{cases}$$

Part A:

In Part A, B and C,

- the value for T_s will be:

$$T_s = 1\ sec \text{ for simplicity of visualization}$$

- the value for time will be:

$$t_{start} = -12\ , \qquad t_{end} = 12, \qquad t_{step} = 0.001,$$

For this part, following y[n] values are determined according to my student ID number.

$$y[-3] = 2, \quad y[-2] = -1, \quad y[-1] = 0, \quad y[0] = 7, \quad y[1] = 0$$

$$y[2] = 3, \quad y[3] = 0, \quad y[4] = 3, \quad y[5] = 0, \quad y[6] = 3$$

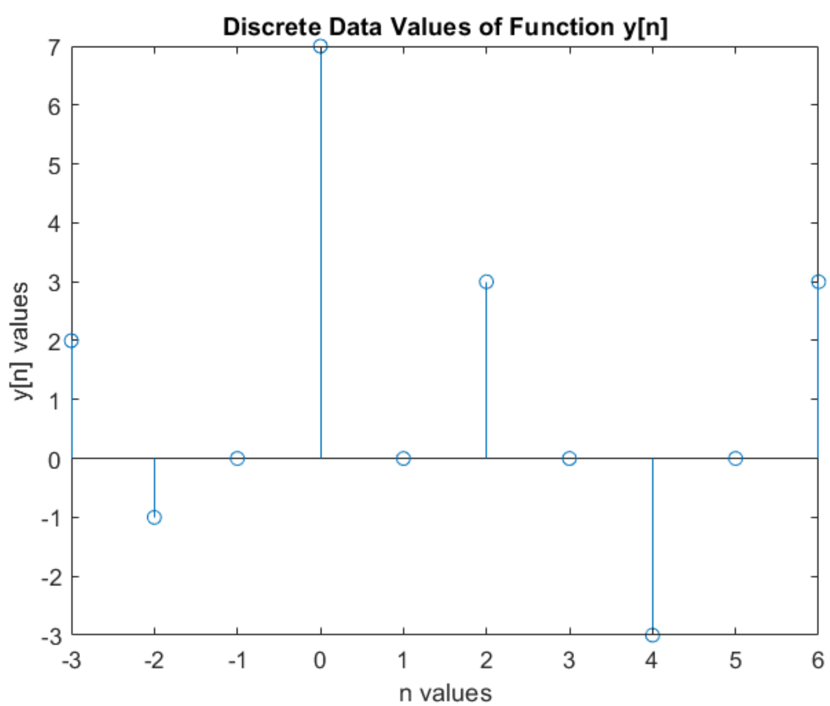

Figure 2: Discrete Sequence Plot For y[n] in Question Part A

Part B:

For this part, following y[n] values are determined according to the following function:

$$y[n] = \begin{cases} 10 \times e^{-0.25|n|}, & -12 \leq n \leq 12 \\ 0, & \text{otherwise} \end{cases}$$

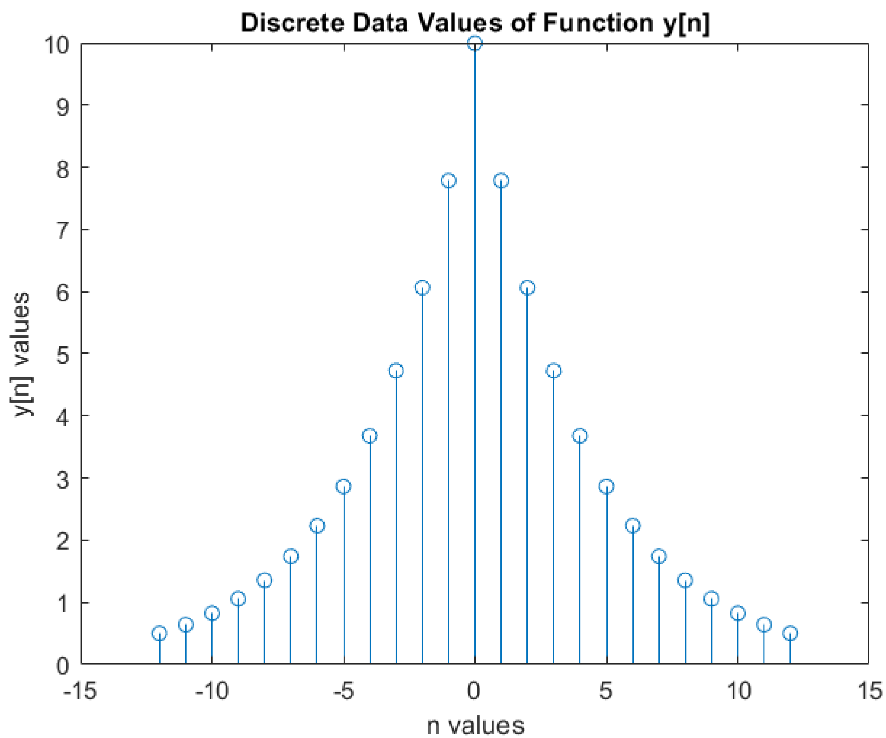

Figure 2: Discrete Sequence Plot For y[n] in Question Part B

Reconstruction According to Pulse Types:

i. Interpolation with Square Pulse

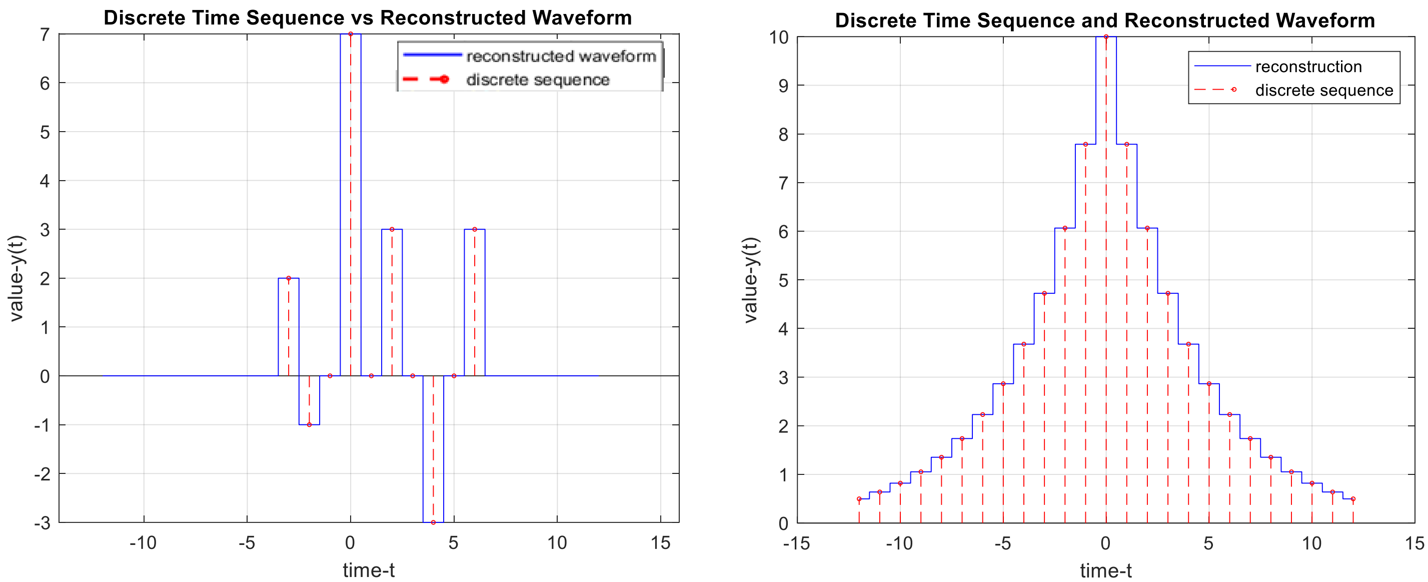

Figure 3:Reconstructed Signal with Square Pulse and Discrete Sequence Values of Part A and B

As square pulse has a width of T_s and amplitude value 1, the sum of each term $y[n]p(t - nT_s)$. Created a flat region having y[n] centered on t, which is nT_s. As can be seen from *Figure 3*, square pulse is a poor approximation, thus, this cannot be an ideal D-to-C conversion.

ii. Interpolation with Triangular Pulse:

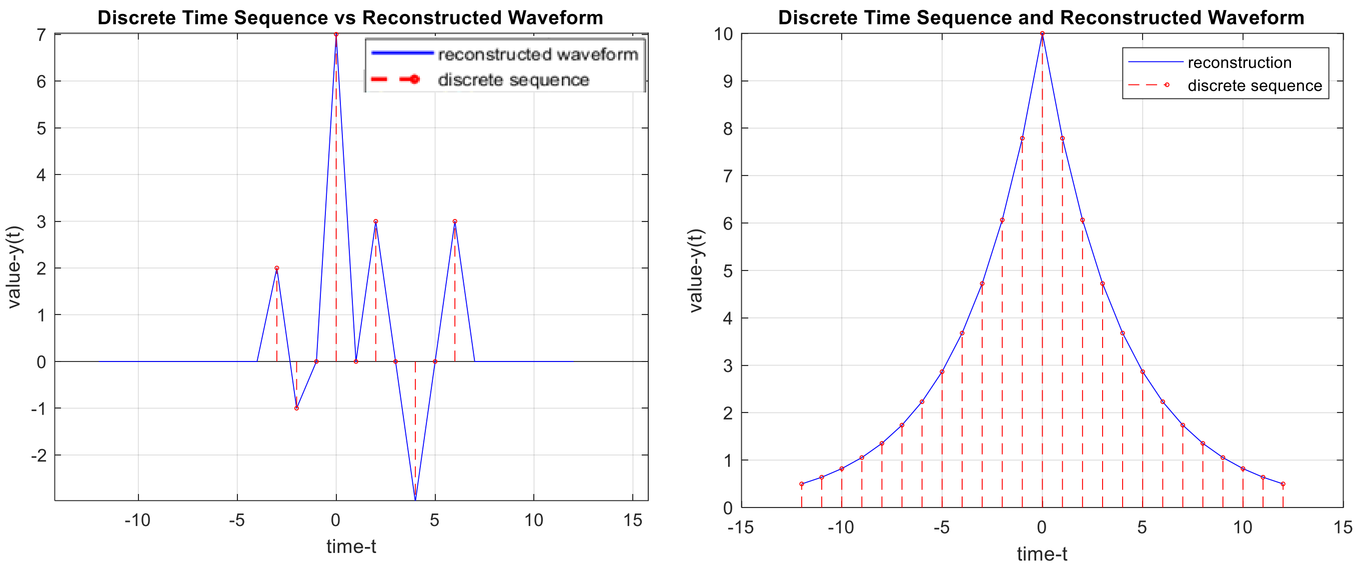

Figure 4: Reconstructed Signal with Triangular Pulse and Discrete Sequence Values of Part A and B

In the case of the pulse type being triangular and having pulse duration 2Ts, no more than two pulses overlap in any time. So, the output, y(t), is a result where the discrete sample points are connected with straight lines. Compared to square pulse, approximation with triangular pulse is a better way to reach ideal D-to-C converter, however, this will also have significant error rates.

iii. Sinc Pulse With Duration $4T_s$

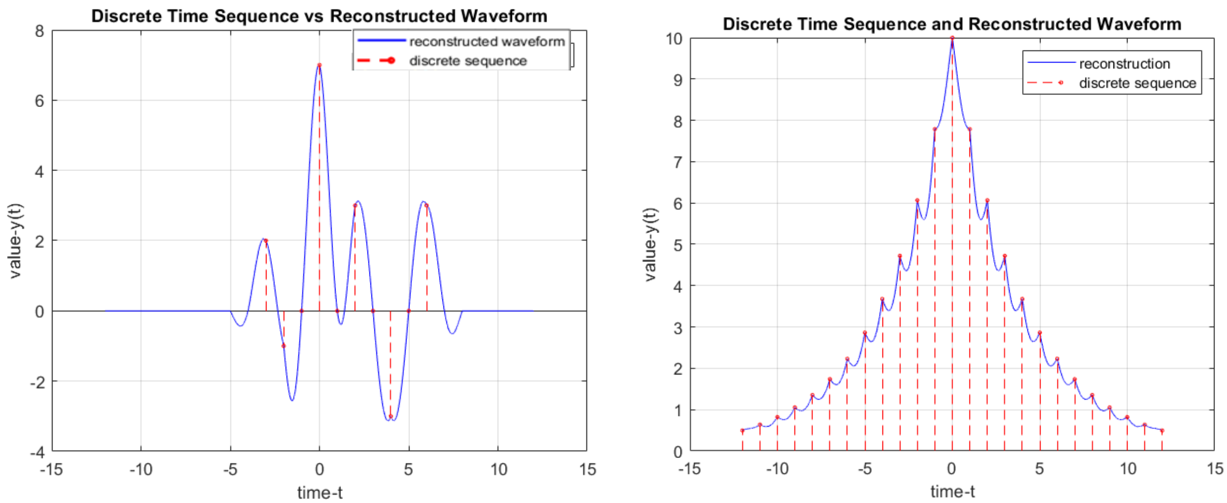

Figure 5: Reconstructed Signal with Sinc Pulse having Duration $2T_s$ and Discrete Sequence Values of Part A and B

The resulting output can be seen from Figure 5. As can be noticed, reconstructed signal became smoother after the experiment of triangular pulse type. Because this pulse type has three parabolicsegments, in each pulse duration four pulses overlap and added to the sum. So, reconstructed signal at time t depends on the two samples before and two samples after.Comparing to the previous reconstructions, this is better, however, it still can be improved.

iv. Truncated Ideal Pulse

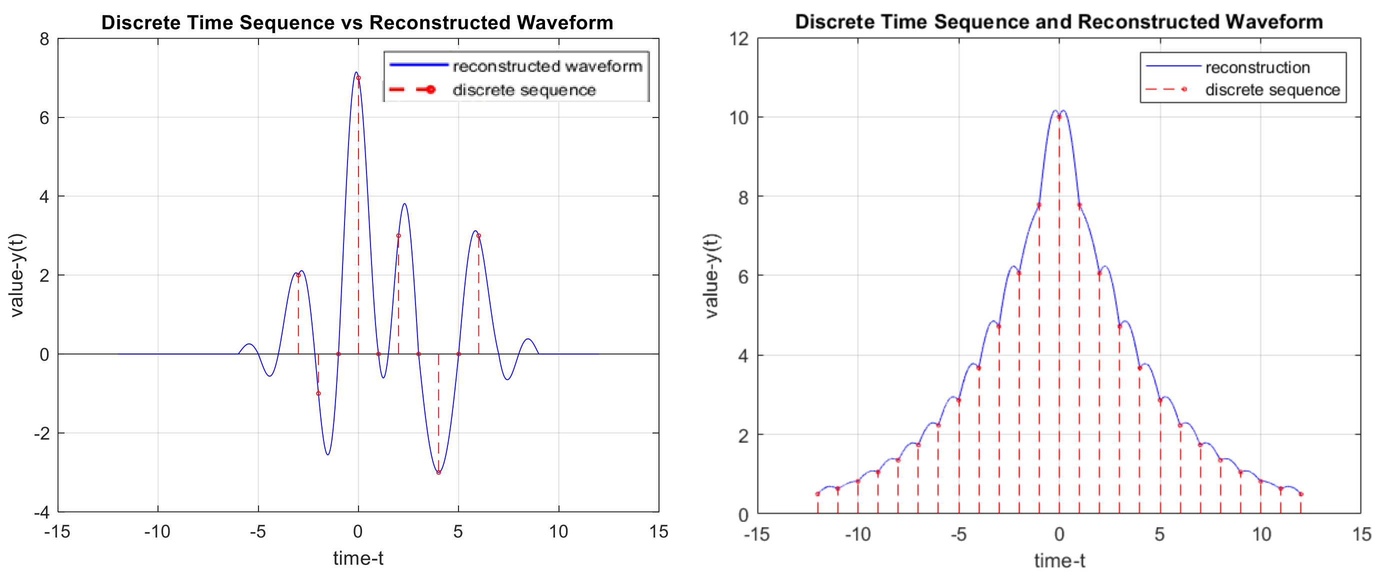

Figure 6: Reconstructed Signal with Truncated Ideal Pulse and Discrete Sequence Values of Part A and B

As can be seen from Figure 6, reconstruction with truncated ideal pulse has given a smoother continuous signal. To compare part A and B, if the original waveform does not have much variance, reconstruction becomes better. Because change is much less for the duration of a pulse, waveform appears to be smoother and accurate in few samples.

Part C:

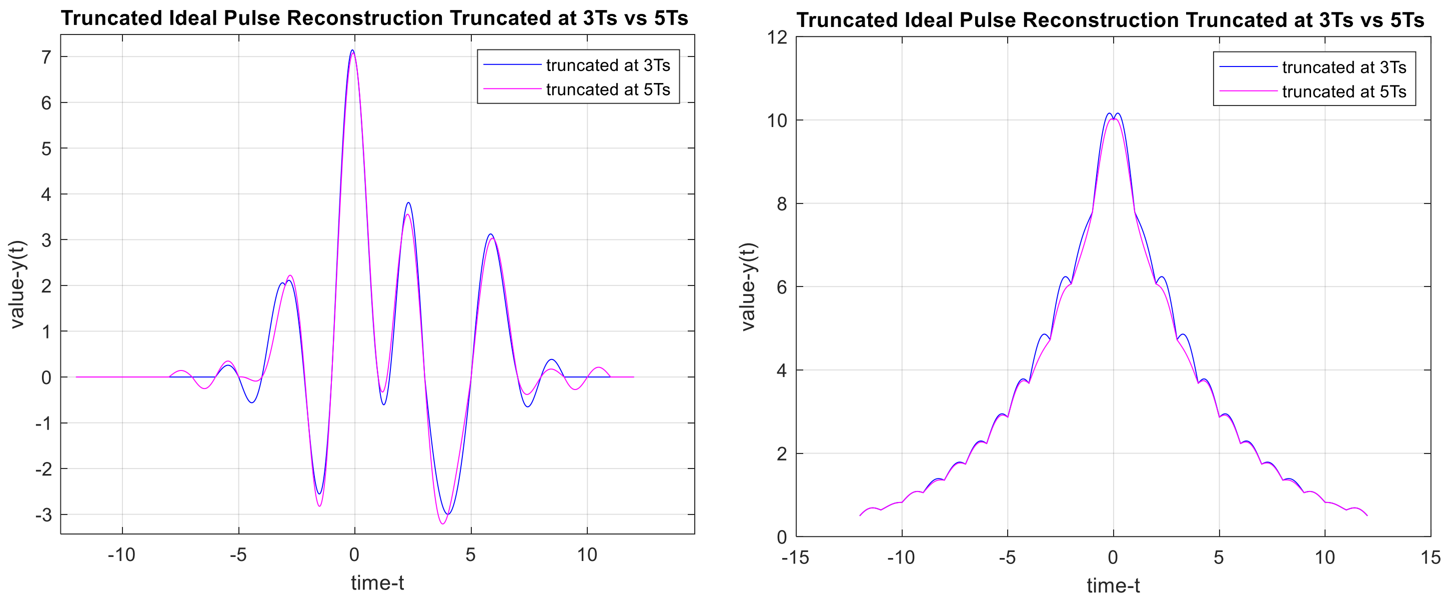

Figure 6: Reconstructed Signals with Truncated Ideal Pulse Truncated at $3T_s$ and $5T_s$ of Part A and B

As can be seen from Figure 6, blue lines are representing sum of truncated ideal pulse signal truncated at $\pm 3T_s$ and magenta lines are representing sum of truncated ideal pulse signal truncated at $\pm 5T_s$. When the duration of the pulse is increased, the sum of pulses became smoother. That is because with pulse having longer duration and smoother structure, reconstruction becomes also smoother. It can be said that sampling rate is a factor for this outcome.

Appendix

```
clc
clear
close all
prompt = "Hello! Choose your pulse for D-C conversion:";
prompt = prompt + newline + "1-Standard square pulse";
prompt = prompt + newline + "2-Standard triangular pulse";
prompt = prompt + newline + "3-Pulse consisting of three parabolic segments";
prompt = prompt + newline + "4-Truncated ideal pulse" + newline;
choice = input(prompt);
prompt_period = "Enter the sampling period: " + newline;
T_s = input(prompt_period); % sampling period Ts
F_s = 1/T_s; %sampling frequency fs


if (choice == 1)
    pulse = @(t, T_s) rectangularPulse(-T_s/2, T_s/2, t);
end
if(choice == 2)
    pulse = @(t, T_s) triangularPulse(-T_s, T_s, t);
end
if (choice == 3)
    pulse = @(t, T_s) ((-2*T_s < t) & (t <=2*T_s)).*(sinc(t/T_s));
end
if(choice == 4)
   pulse = @(t, T_s) ((-3*T_s < t) & (t <= 3*T_s)).*(sinc(t/T_s));
end
```

```
%Create signal for part a
n_a=-3:6;
yn_a = [2,-1,0,7,0,3,0,-3,0,3];
% stem (n_a, yn_a)
% xlabel('n values');
% ylabel('y[n] values');
% title('Discrete Data Values of Function y[n]');
%Create signal for part b
n_b = -12:12;
yn_b = 10*exp(-abs(n_b)/4);
% stem (n_b, yn_b)
% xlabel('n values');
% ylabel('y[n] values');
% title('Discrete Data Values of Function y[n]');
```

```
prompt_example = "Enter example a, b or c: " + newline;
example = input(prompt_example, 's');
t = -12:0.001:12;
if( example == 'a')
    rc = zeros(length(yn_a),length(t));
    for n = n_a
        rc(n+4, :) = yn_a(n+4)*pulse(t-n*T_s, T_s);
    end
    plot(t, sum(rc), 'color', 'b');hold on
    legend('reconstructed waveform');
    stem(n_a,yn_a, 'LineStyle', '--', 'Color','red', 'MarkerSize',2);hold off
    xlabel('time-t');
    ylabel('value-y(t)');
    title('Discrete Time Sequence vs Reconstructed Waveform');
    legend('reconstructed waveform');
    grid
end
```

```
if ( example == 'b')
    rc = zeros(length(yn_b),length(t));
    for n = n_b
        rc(n+13, :) = yn_b(n+13)*pulse(t-n*T_s, T_s);
    end
    plot(t, sum(rc), 'color', 'b');hold on
    stem(n_b,yn_b, 'LineStyle', '--', 'Color','red', 'MarkerSize',2);hold off
    xlabel('time-t');
    ylabel('value-y(t)');
    title('Discrete Time Sequence and Reconstructed Waveform');
    legend({'reconstruction','discrete sequence'});
    grid
end
```

```
if ( example == 'c')
    pulse_3 = @(t, T_s) ((-3*T_s <= t) & (t <= 3*T_s)).*(sinc(t/T_s));
    pulse_5 = @(t, T_s) ((-5*T_s <= t) & (t <= 5*T_s)).*(sinc(t/T_s));
    %For Part A
    rc_3 = zeros(length(yn_a),length(t));
    for n = n_a
        rc_3(n+4, :) = yn_a(n+4)*pulse_3(t-n*T_s, T_s);
    end
    rc_5 = zeros(length(yn_a),length(t));
    for n = n_a
        rc_5(n+4, :) = yn_a(n+4)*pulse_5(t-n*T_s, T_s);
```

```
    end
    figure
    plot(t, sum(rc_3),'b' , t, sum(rc_5), 'm');
    legend({'truncated at 3Ts','truncated at 5Ts'});
    xlabel('time-t');
    ylabel('value-y(t)');
    title('Truncated Ideal Pulse Reconstruction Truncated at 3Ts vs 5Ts');
    %legend('truncated at 3Ts','truncated at 5Ts');
    grid
    %For Part B
    rc_3 = zeros(length(yn_b),length(t));
    for n = n_b
        rc_3(n+13, :) = yn_b(n+13)*pulse_3(t-n*T_s, T_s);
    end
    rc_5 = zeros(length(yn_b),length(t));
    for n = n_b
        rc_5(n+13, :) = yn_b(n+13)*pulse_5(t-n*T_s, T_s);
    end
    figure; plot(t, sum(rc_3), 'b',t, sum(rc_5), 'm')
    legend({'truncated at 3Ts','truncated at 5Ts'});
    xlabel('time-t');
    ylabel('value-y(t)');
    title('Truncated Ideal Pulse Reconstruction Truncated at 3Ts vs 5Ts');
    grid
end
```