ECEN 220 Lab 5

Sampling & up-scaling

Niels Clayton
300437590

October 6^{th}, 2019

---

1 Square wave generation & zero insertion

Generate a square wave of the following form:

```
function [x] = square_wave(f_0, sum_lower, sum_upper)
    syms q t
    func = (sin(2.*pi.*(2.*q + 1).* f_0.*t))./(2.*q +1);
    x(t) = symsum(func, q, sum_lower, sum_upper) .*(4/pi);
end
```

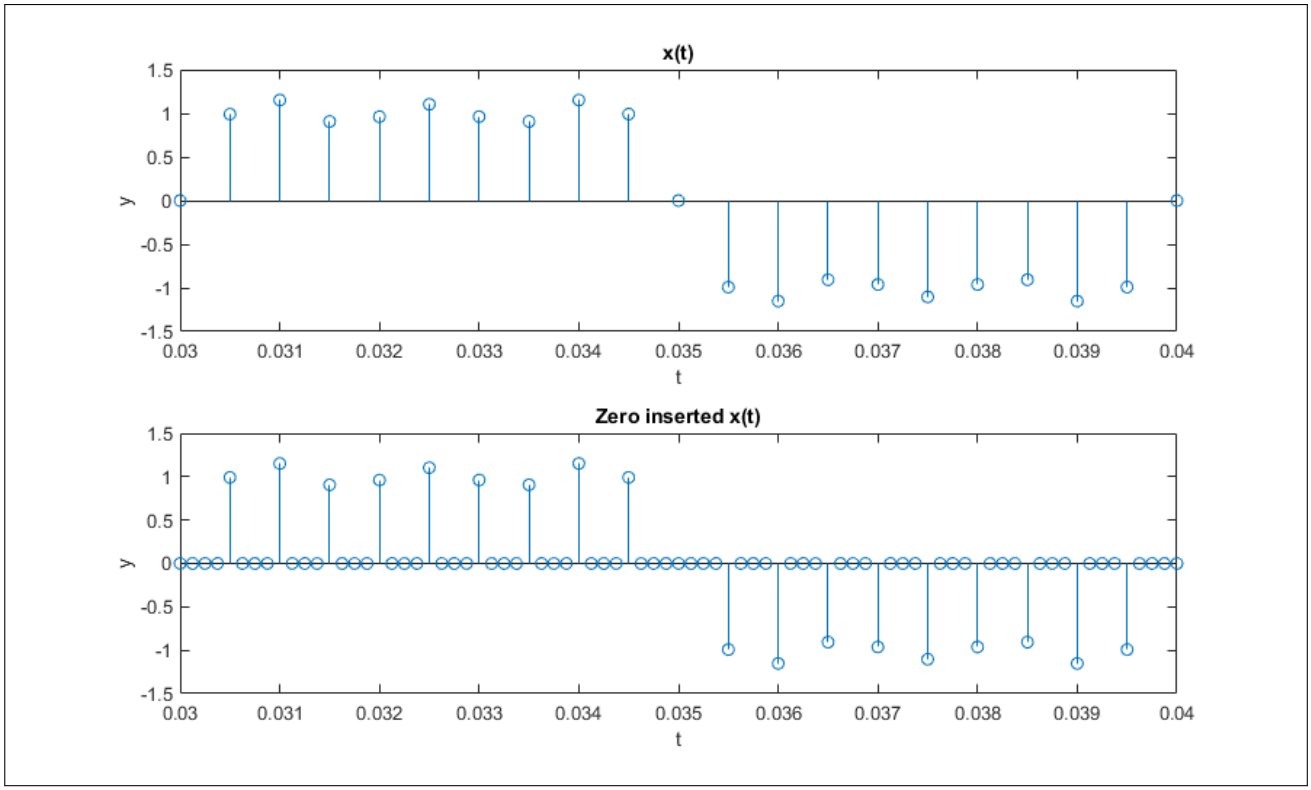

Figure 1: Original $x[n]$ & zero inserted $x[n]$

Using the matlab function `upsample()` we up-sample the square wave by a factor of 4 by inserting zero values between all the data points.

2 Low pass filter generation & it's frequency response

Generate a low pass filter of the following form:

```
function [h, n] = lowpass(L, M)
    n = -L:L;
    h = M.*(sin((pi.*n)./M))./(pi.*n);
    h(L+1) = 1;
end
```

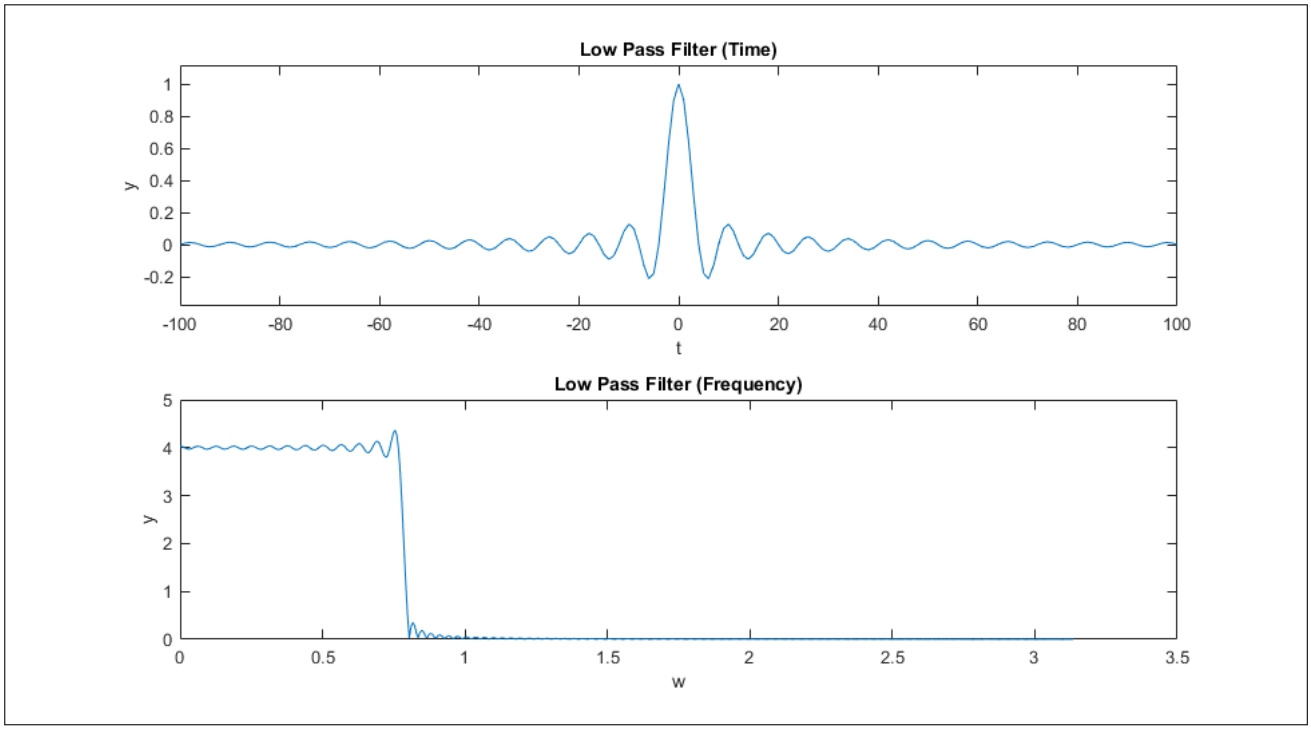

Figure 2: Low-Pass filter and its frequency response

In figure 2 it can be seen that the low pass filter in time is a *sinc* function, and a square pulse in frequency. However only the positive half of the square pulse can be observed due to the nature of the Matlab `freqz()` function, however we know that it will be symmetrical around the 0 point.

After generating this low-pass filter, we will filter the up-sampled square wave with it using the Matlab function `filter()` the result of which can be seen below in figure 3, and a phase shifted version can be seen in figure 4 for easier comparisons.

3 Compare the original signal to the up-sampled and filtered signal

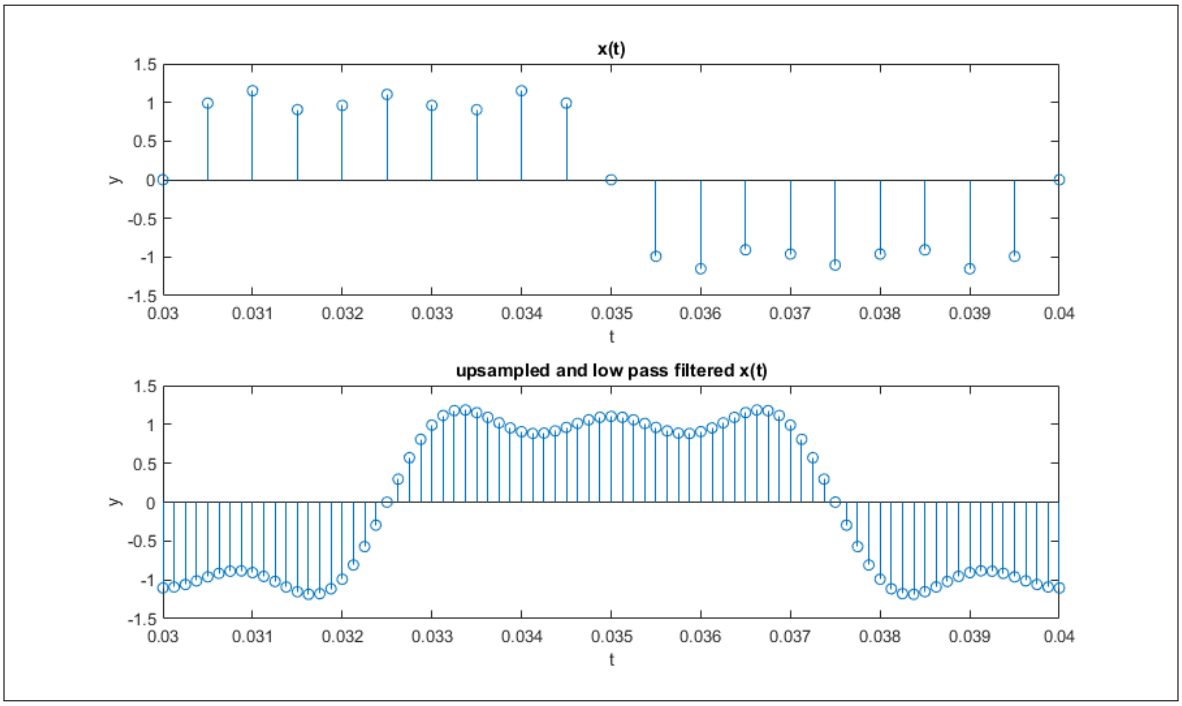

Figure 3: Original signal compared to the up-sampled signal

4 Phase shift the up-sampled and filtered signal

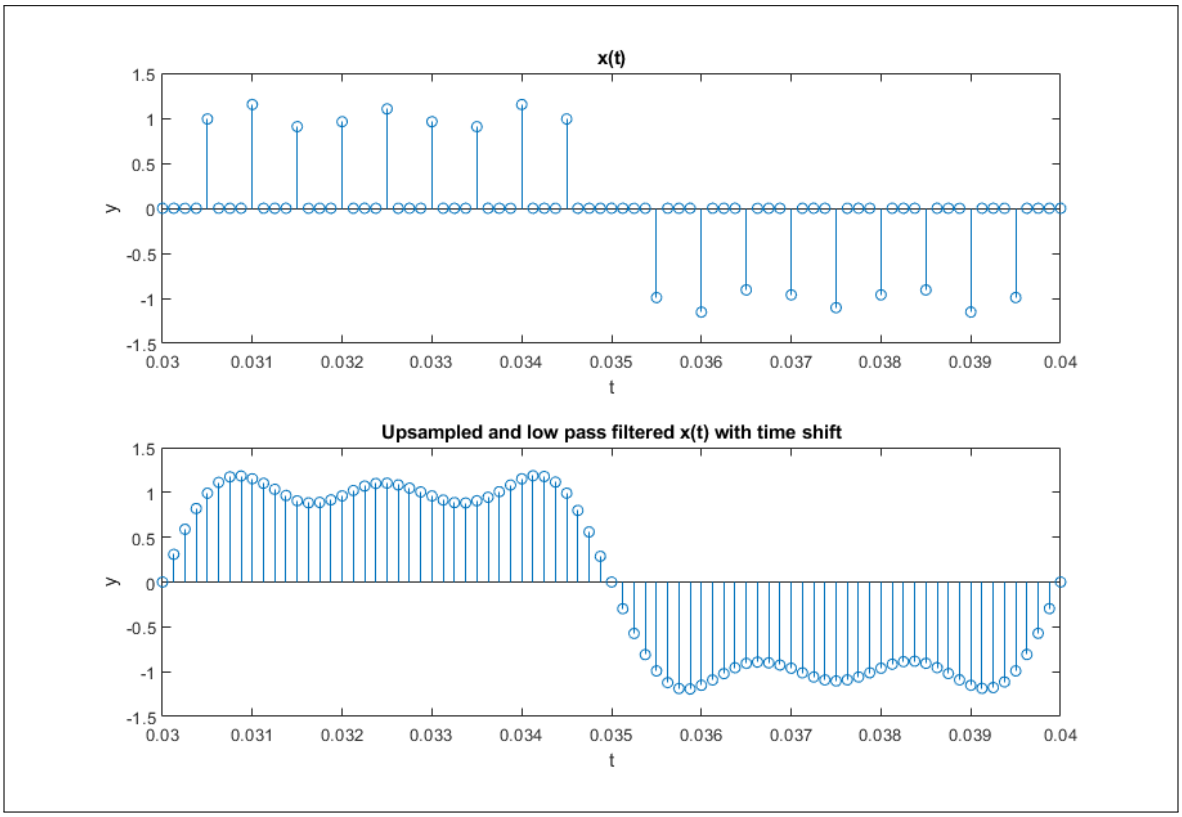

Figure 4: Up-sampled signal phase shifted for easier comparison

It can be seen that the up-sampled signal is very close if not identical to the information held within the original signal.

```
%% 1
clc, clear;

f_0 = 100;
M = 4;

x = square_wave(f_0, 0, 2);

sample_f = 2000;
upsample_f = M*sample_f;

T = 1/sample_f;
T1 = 1/upsample_f;

t = (0:sample_f-1).*T;
t1 = (0:upsample_f-1).*T1;

y = double(x(t));
y1 = upsample(y, M);

subplot(2,1,1);
stem(t, y)
xlim([30e-3 40e-3])
title("x(t)")
xlabel('t')
ylabel('y')

subplot(2,1,2);
stem(t1, y1)
xlim([30e-3 40e-3])
title("Zero inserted x(t)")
xlabel('t')
ylabel('y')

%% 2
clc
%generate sinc function
L = 100;
[h, n] = lowpass(L, M);
[H, W] = freqz(h, 1, 500);

subplot(2,1,1);
plot(n, h)
title("Low Pass Filter (Time)")
xlabel('t')
ylabel('y')


subplot(2,1,2);
plot(W, abs(H))
title("Low Pass Filter (Frequency)")
xlabel('w')
ylabel('y')

%% 3
```

```
clc

filtered = filter(h, 1, y1);

%% 4

subplot(2,1,1);
stem(t, y)
xlim([30e-3 40e-3])
title("x(t)")
xlabel('t')
ylabel('y')

subplot(2,1,2);
stem(t1, filtered)
xlim([30e-3 40e-3])
title("upsampled and low pass filtered x(t)")
xlabel('t')
ylabel('y')

%% 5
clc

delay_filter = zeros(1,L+1);
delay_filter(L+1) = -1;

delayed = filter(delay_filter, 1, filtered);


subplot(2,1,1);
stem(t1, y1)
xlim([30e-3 40e-3])
title("x(t)")
xlabel('t')
ylabel('y')

subplot(2,1,2);
stem(t1, delayed)
xlim([30e-3 40e-3])
title("Upsampled and low pass filtered x(t) with time shift")
xlabel('t')
ylabel('y')
```