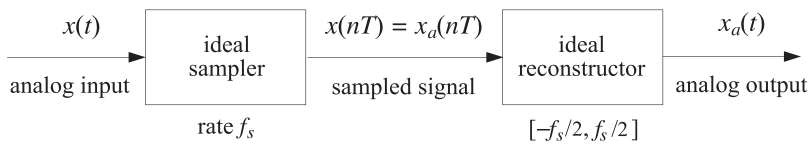
# Lab 4 – Sampling, Aliasing, FIR Filtering

This is a software lab. In your report, please include all Matlab code, numerical results, plots, and your explanations of the theoretical questions. The due date is *one week* from assignment.

## 4.1. Sampling and Aliasing – Sinusoids

The aim of this lab is to demonstrate the effects of aliasing arising from improper sampling. A given analog signal $x(t)$ is sampled at a rate $f_s$, the resulting samples $x(nT)$ are then reconstructed by an *ideal* reconstructor into the analog signal $x_a(t)$. Improper choice of $f_s$ will result in a different signal, $x_a(t) \neq x(t)$, even though the two agree at their sample values, that is, $x_a(nT) = x(nT)$. The procedure is illustrated in the following figure:




**Lab Procedure**

a. Consider an analog signal $x(t)$ consisting of three sinusoids of frequencies of 1 kHz, 4 kHz, and 6 kHz:

$$x(t) = \sin(2\pi t) + 2\sin(8\pi t) + 3\sin(12\pi t)$$

where $t$ is in milliseconds. Show that if this signal is sampled at a rate of $f_s = 5$ kHz, it will be aliased with the following signal, in the sense that their sample values will be the same:

$$x_a(t) = 2\sin(2\pi t)$$

On the same graph, plot the two signals $x(t)$ and $x_a(t)$ versus $t$ in the range $0 \le t \le 2$ msec. To this plot, add the time samples $x(t_n)$ and verify that $x(t)$ and $x_a(t)$ intersect precisely at these samples.

b. Repeat part (a) with $f_s = 10$ kHz. In this case, determine the signal $x_a(t)$ with which $x(t)$ is aliased. Plot both $x(t)$ and $x_a(t)$ on the same graph over the same range $0 \le t \le 2$ msec. Verify again that the two signals agree at the sampling instants, $x_a(nT) = x(nT)$. See example graphs at the end.

## 4.2. Sampling and Aliasing – Square Wave

Consider a periodic pulse wave $x(t)$ with period $T_0 = 1$ sec, as shown below. Let $p(t)$ denote one basic period of $x(t)$ defined over the time interval $0 \le t \le 1$:

$$p(t) = \begin{cases} 1, & \text{if} \quad 0.125 < t < 0.375 \\ -1, & \text{if} \quad 0.625 < t < 0.875 \\ 0.5, & \text{if} \quad t = 0.125 \quad \text{or} \quad t = 0.375 \\ -0.5, & \text{if} \quad t = 0.625 \quad \text{or} \quad t = 0.875 \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

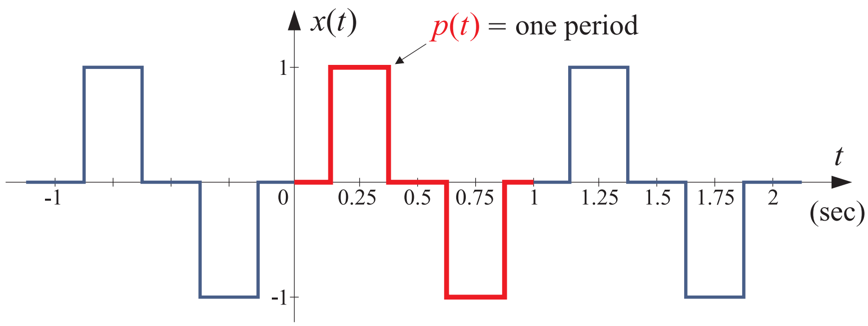
This periodic signal admits a Fourier series expansion containing only sine terms with odd harmonics of the basic period $f_0 = 1/T_0 = 1$ Hz, that is, the frequencies $f_m = mf_0$, $m = 1, 3, 5, \ldots$ Hz:

$$x(t) = \sum_{m=1,3,5,\ldots} b_m \sin(2\pi m t) = b_1 \sin(2\pi t) + b_3 \sin(6\pi t) + b_5 \sin(10\pi t) + \cdots \tag{4.2}$$

The Fourier series coefficients are given as follows, for $m = 1, 3, 5, 7, \ldots$

$$b_m = \frac{\cos(\pi m/4) - \cos(3\pi m/4) - \cos(5\pi m/4) + \cos(7\pi m/4)}{\pi m}$$

The reason why the signal $x(t)$ was defined to have the values $\pm 0.5$ at the discontinuity points is a consequence of a theorem that states that any finite sum of Fourier series terms will always pass through the mid-points of discontinuities.




**Lab Procedure**

a. Define the function of Eq. (4.1) in MATLAB using a one-line anonymous function definition of the form:

```
p = @(t) ...        % one period of the square wave
```

using vectorized relational operations, such as, `(0.125<t & t<0.375)`.

b. To understand the nature of the approximation of the square wave by the Fourier series sum, truncate the sum to a finite number of terms, that is, with $M$ odd,

$$x_M(t) = \sum_{m=1,3,5,\dots}^{M} b_m \sin(2\pi m t) = b_1 \sin(2\pi t) + b_3 \sin(6\pi t) + \cdots + b_M \sin(2\pi M t) \tag{4.3}$$

Evaluate and plot $x(t)$ and $x_M(t)$ over one period $0 \le t \le 1$, for $M = 21$ and $M = 41$.

c. The pulse waveform $x(t)$ is now sampled at the rate of $f_s = 8$ Hz and the resulting samples $x(nT)$ are reconstructed by an *ideal* reconstructor resulting into the aliased analog signal $x_a(t)$.

The spectrum of the sampled signal consists of the periodic replication of the harmonics of $x(t)$ at multiples of $f_s$. Because $f_s/f_0 = 8$ is an even integer, all the odd harmonics that lie outside the Nyquist interval, $[-4, 4]$ Hz, will be wrapped onto the odd harmonics that lie inside this interval, that is, onto $\pm 1, \pm 3$ Hz. This can be verified by listing a few of the odd harmonics of $x(t)$ and the corresponding wrapped ones modulo $f_s$ that lie within the Nyquist interval:

```
1   3   5   7   9   11   13   15   17   19   21   23   25   27   29 ...
1   3  -3  -1   1    3   -3   -1    1    3   -3   -1    1    3   -3 ...
```

where the bottom row is obtained by subtracting enough multiples of $f_s = 8$ from each harmonic until it is brought to lie within the interval $[-4, 4]$ Hz. This means then that the aliased signal will consist only of sinusoids of frequencies $f_1 = 1$ and $f_3 = 3$ Hz,

$$x_a(t) = A \sin(2\pi t) + B \sin(6\pi t) \tag{4.4}$$

Determine the coefficients $A, B$ by setting up two equations in the two unknowns $A, B$ by enforcing the matching equations $x_a(nT) = x(nT)$ at the two sampling instants $n = 1, 2$.

On the same graph, plot one period of the pulse wave $x(t)$ together with $x_a(t)$. Verify that they agree at the eight sampling time instants that lie within this period. Because of the sharp transitions of the square wave, you must use a very dense time vector, for example,

```
t = linspace(0,1,4097);
```

Also, if you wish, you may do part (c) and part (d), as special cases of part (e).

d. Assume, next, that the pulse waveform $x(t)$ is sampled at the rate of $f_s = 16$ Hz. By considering how the out-of band harmonics wrap into the Nyquist interval $[-8, 8]$ Hz, show that now the aliased signal $x_a(t)$ will have the form:

$$x_a(t) = a_1 \sin(2\pi t) + a_2 \sin(6\pi t) + a_3 \sin(10\pi t) + a_4 \sin(14\pi t)$$

where the coefficients $a_i$ are obtained by the condition that the signals $x(t)$ and $x_a(t)$ agree at the first four sampling instants $t_n = nT = n/16$ Hz, for $n = 1, 2, 3, 4$. These four conditions can be arranged into a 4×4 matrix equation of the form:

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} * \\ * \\ * \\ * \end{bmatrix}$$

Determine the numerical values of the starred entries. Then, using MATLAB, solve this matrix equation for the coefficients $a_i$. Once $a_i$ are known, the signal $x_a(t)$ is completely defined.

On the same graph, plot one period of the pulse waveform $x(t)$ together with $x_a(t)$. Verify that they agree at the 16 sampling time instants that lie within this period.

e. The methods of parts (c,d) can be generalized to any sampling rate $f_s$ such that $L = f_s/f_0$ is an even integer (so that all the out-of-band odd harmonics will wrap onto the odd harmonics within the Nyquist interval). First show that the number of odd harmonics within the positive side of the Nyquist interval is:

$$K = \text{floor}\left(\frac{L+2}{4}\right)$$

This means that the aliased signal will be the sum of $K$ terms:

$$x_a(t) = \sum_{k=1}^{K} a_k \sin\big(2\pi(2k-1)t\big) \tag{4.5}$$

By matching $x_a(t)$ to $x(t)$, or $p(t)$, at the first $K$ sampling instants $n = 1, 2, \dots, K$, set up a linear system of $K$ equations in the $K$ unknowns $a_k$, i.e., with $t_n = nT$,

$$\sum_{k=1}^{K} a_k \sin\big(2\pi(2k-1)t_n\big) = p(t_n)\,, \quad n = 1, 2, \dots, K \tag{4.6}$$

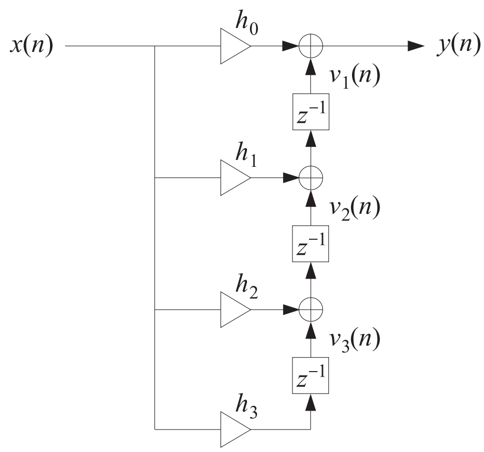
and solve it with Matlab. Once you have the coefficients $a_k$, evaluate and plot $x(t)$ and $x_a(t)$, and add the sampled points on the graph. Repeat this for the following eight, progressively larger, sampling rates:

$$f_s = [4,\ 8,\ 16,\ 24,\ 32,\ 40,\ 48,\ 64]$$

It should be evident that even though the square wave is not a bandlimited signal, it can still be sampled adequately if the sampling rate is chosen to be large enough.

## 4.3. FIR Filtering

The objective of this lab is to implement your own version of the built-in function **filter** adapted to FIR filters. The IIR case will be considered in a future lab. The documentation for **filter** states that it is implemented using the transposed block diagram realization. For example, for an order-3 FIR filter that realization and the system of difference equations implementing it are:




$$\boxed{\begin{aligned} y(n) &= h_0\,x(n) + v_1(n) \\ v_1(n+1) &= h_1\,x(n) + v_2(n) \\ v_2(n+1) &= h_2\,x(n) + v_3(n) \\ v_3(n+1) &= h_3\,x(n) \end{aligned}}$$

For an $M$th order filter, the computational algorithm is,

$$\boxed{\begin{aligned} y(n) &= h_0\,x(n) + v_1(n) \\ v_1(n+1) &= h_1\,x(n) + v_2(n) \\ v_2(n+1) &= h_2\,x(n) + v_3(n) \\ &\ \vdots \\ v_{M-1}(n+1) &= h_{M-1}\,x(n) + v_M(n) \\ v_M(n+1) &= h_M\,x(n) \end{aligned}} \qquad \mathbf{v}(n) = \begin{bmatrix} v_1(n) \\ v_2(n) \\ \vdots \\ v_M(n) \end{bmatrix} = \text{state vector}$$

The state vector $\mathbf{v}(n)$ represents the current contents of the $M$ delays. The algorithm uses the current state $\mathbf{v}(n)$ to compute the current output $y(n)$ from the current input $x(n)$, and then, it updates the state to the next time instant, $\mathbf{v}(n+1)$. Usually, the state vector is initialized to zero, but it can be initialized to an arbitrary vector, say, $\mathbf{v}_{\text{init}}$.

**Lab Procedure**

a. Write a MATLAB function, say, **firtr.m**, that implements the above algorithm and has the possible syntaxes:

```
y = firtr(h,x);
[y,vout] = firtr(h,x,vin);

%   h  = (M+1)-dimensional filter vector (row or column)
%   x  = length-N vector of input samples (row or column)
%   y  = length-N vector of output samples (row or column)
% vin  = M-dimensional vector of initial states - zero vector, by default
% vout = M-dimensional final state vector, i.e., final contents of delays
```

Test your function with the following case:

```
x = [1, 1, 2, 1, 2, 2, 1, 1];
h = [1, 2, -1, 1];
y = [1, 3, 3, 5, 3, 7, 4, 3]    % expected result
```

b. The function **firtr** can be run on a sample by sample basis to generate the successive internal state vectors, for example, using a loop such as,

```
v = zeros(1,M);                      % initial state vector
for n=1:length(x)
   [y(n),vout] = firtr(h,x(n),v);    % recycled state vector v
   v = vout;                         % next state
end
```

Add appropriate **fprintf** commands before, within, and after this loop to generate the following table of values for the above example,

```
 n    x    y    v1   v2   v3
----------------------------
 0    1    1    0    0    0
 1    1    3    2   -1    1
 2    2    3    1    0    1
 3    1    5    4   -1    2
 4    2    3    1    1    1
 5    2    7    5   -1    2
 6    1    4    3    0    2
 7    1    3    2    1    1
 8    -    -    3    0    1
```

where $v_1, v_2, v_3$ are the internal states at each time instant.

c. Write a function, **myconv.m**, that uses the above function **firtr** to implement the convolution of two vectors **h**, **x**. It should be functionally equivalent to the built-in function **conv** and have usage,

```
% y = myconv(h,x);
%
% h  = (M+1)-dimensional filter vector (row or column)
% x  = length-N vector of input samples (row or column)
% y  = length-(N+M) vector of output samples (row or column)
```

Test it on the following case:

```
x = [1, 1, 2, 1, 2, 2, 1, 1];
h = [1, 2, -1, 1];
y = [1, 3, 3, 5, 3, 7, 4, 3, 3, 0, 1]    % expected result
```

## 4.4. Filtering of Noisy Signals

A length-$N$ signal $x(n)$ is the sum of a desired signal $s(n)$ and interference $v(n)$:

$$x(n) = s(n) + v(n)\,, \quad 0 \le n \le N-1$$

where

$$s(n) = \sin(\omega_0 n)\,, \quad v(n) = \sin(\omega_1 n) + \sin(\omega_2 n)\,, \quad 0 \le n \le N-1$$

with

$$\omega_1 = 0.1\pi\,, \quad \omega_0 = 0.2\pi\,, \quad \omega_2 = 0.3\pi \qquad \text{[radians/sample]}$$

In order to remove $v(n)$, the signal $x(n)$ is filtered through a bandpass FIR filter that is designed to pass the frequency $\omega_0$ and reject the interfering frequencies $\omega_1, \omega_2$. An example of such a filter of order

$M = 150$ can be designed with the Fourier series method using a Hamming window, and has impulse response:

$$h(n) = w(n)\left[\frac{\sin(\omega_b(n - M/2)) - \sin(\omega_a(n - M/2))}{\pi(n - M/2)}\right], \quad 0 \le n \le M$$

where $\omega_a = 0.15\pi$, $\omega_b = 0.25\pi$, and $w(n)$ is the Hamming window:

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{M}\right), \quad 0 \le n \le M$$

It has an effective passband $[\omega_a, \omega_b] = [0.15\pi, 0.25\pi]$. To avoid a computational issue at $n = M/2$, you may use MATLAB's built-in function **sinc**, which is defined as follows:

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$$

**Lab Procedure**

a. Let $N = 200$. On the same graph plot $x(n)$ and $s(n)$ versus $n$ over the interval $0 \le n \le N - 1$.

b. Filter $x(n)$ through the filter $h(n)$ using your function **firtr**, and plot the filtered output $y(n)$, together with $s(n)$, for $0 \le n \le N - 1$. Apart from an overall delay introduced by the filter, $y(n)$ should resemble $s(n)$ after the $M$ initial transients.

c. To see what happened to the interference, filter the signal $v(n)$ separately through the filter and plot the output, on the same graph with $v(n)$ itself.

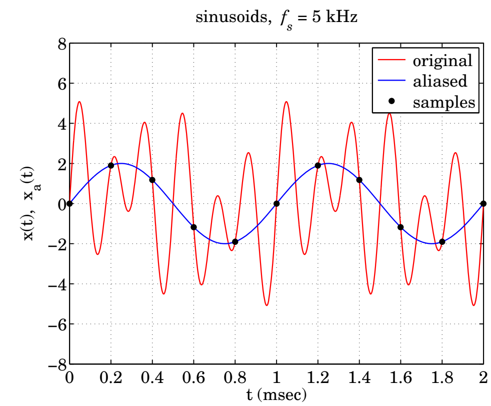
d. Using the built-in MATLAB function **freqz** calculate and plot the magnitude response of the filter over the frequency interval $0 \le \omega \le 0.4\pi$:

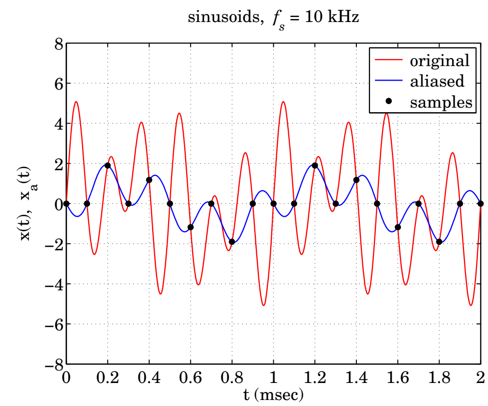
$$|H(\omega)| = \left|\sum_{n=0}^{M} h(n)e^{-j\omega n}\right|$$

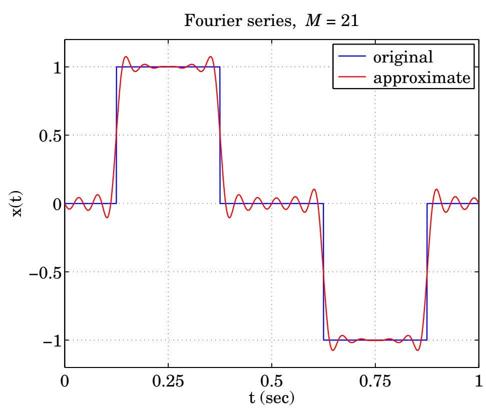
Indicate on that graph the frequencies $\omega_1, \omega_0, \omega_2$. Repeat the plot of $|H(\omega)|$ in dB units.

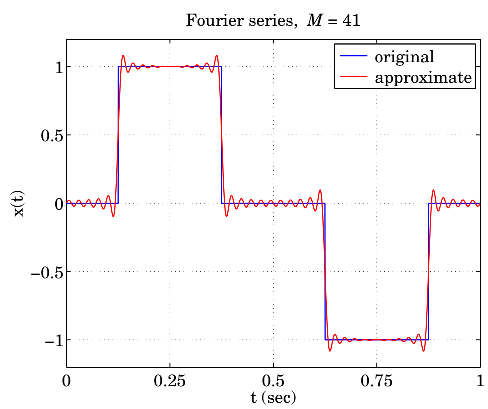
e. Redesign the filter with $M = 200$ and repeat parts (a)–(d). Discuss the effect of choosing a longer filter length.

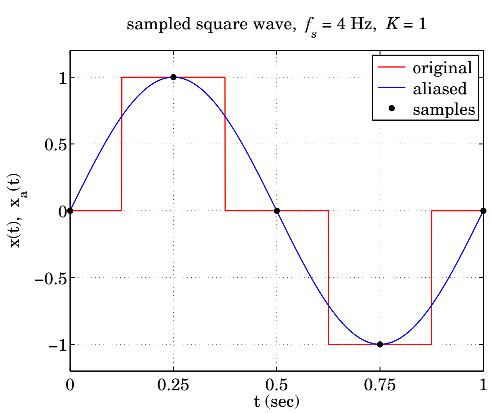
## Example Graphs

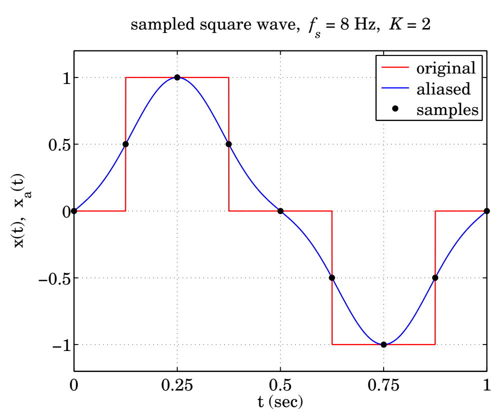


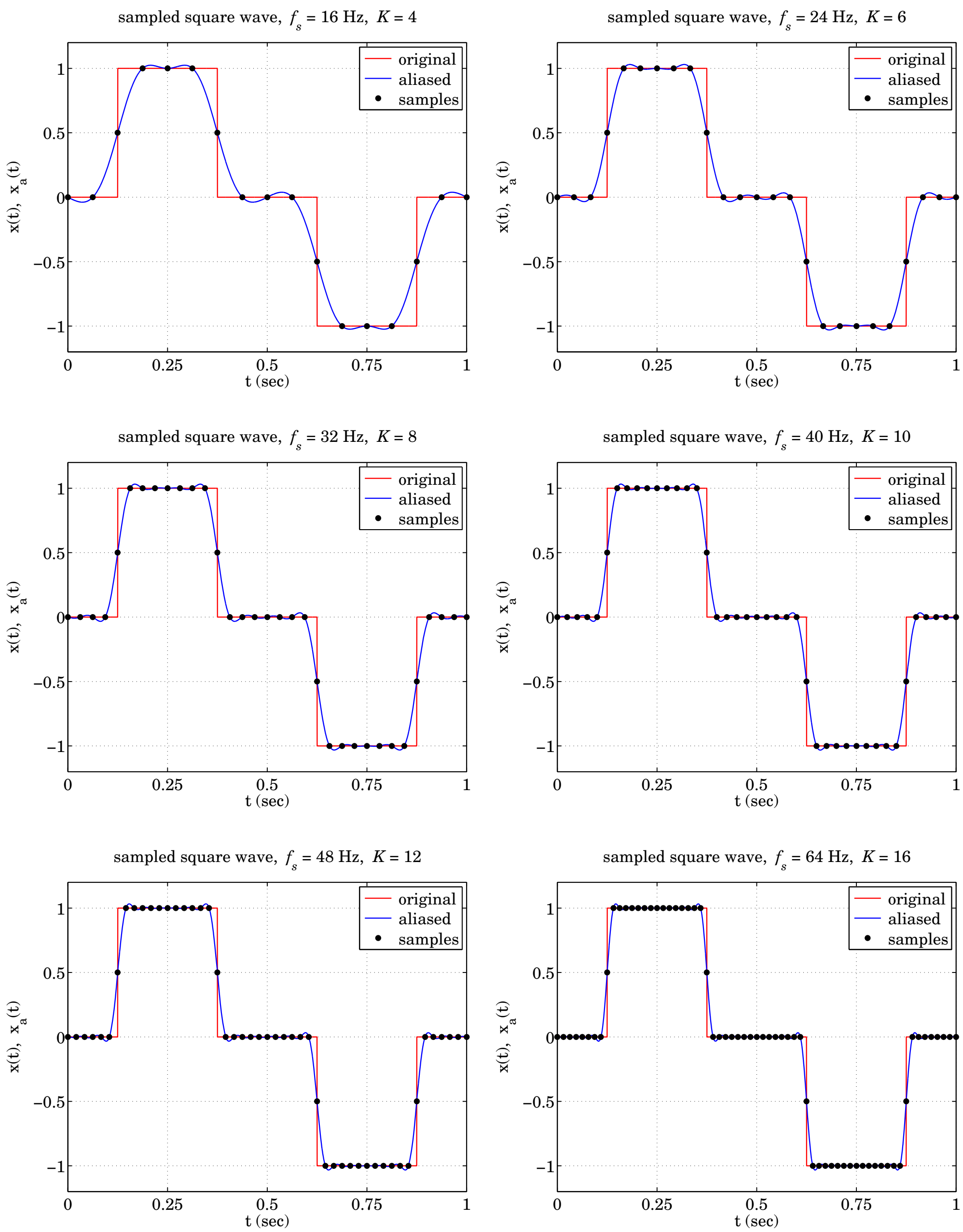



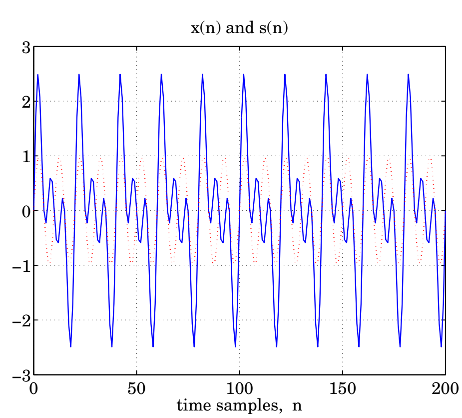



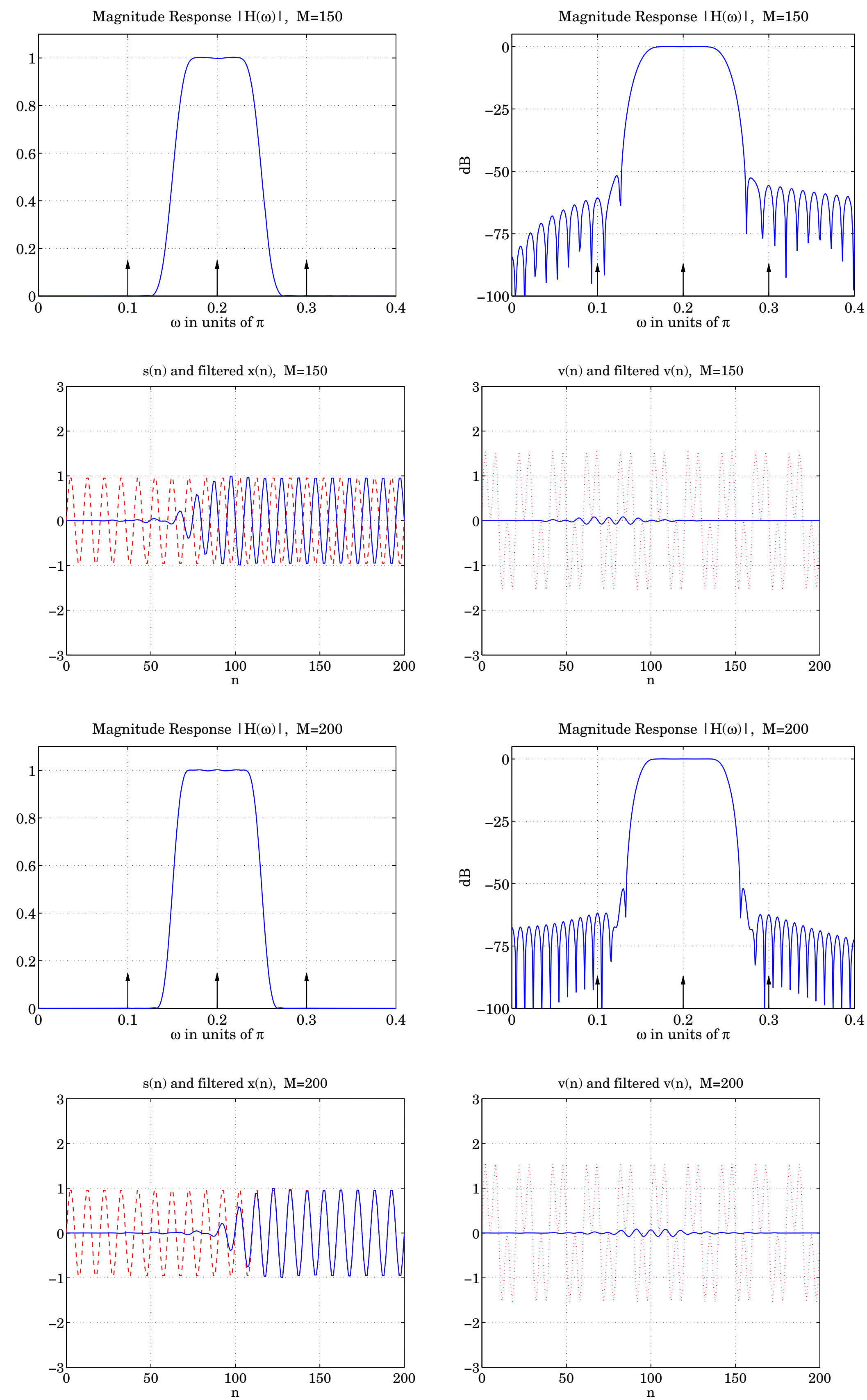




sampled square wave, $f_s$ = 16 Hz, $K$ = 4
sampled square wave, $f_s$ = 24 Hz, $K$ = 6
sampled square wave, $f_s$ = 32 Hz, $K$ = 8
sampled square wave, $f_s$ = 40 Hz, $K$ = 10
sampled square wave, $f_s$ = 48 Hz, $K$ = 12
sampled square wave, $f_s$ = 64 Hz, $K$ = 16
original
aliased
samples
$x(t)$, $x_a(t)$
t (sec)


x(n) and s(n)
time samples, n


Magnitude Response |H(ω)|, M=150
ω in units of π
Magnitude Response |H(ω)|, M=150
dB
ω in units of π
s(n) and filtered x(n), M=150
n
v(n) and filtered v(n), M=150
n
Magnitude Response |H(ω)|, M=200
ω in units of π
Magnitude Response |H(ω)|, M=200
dB
ω in units of π
s(n) and filtered x(n), M=200
n
v(n) and filtered v(n), M=200
n