Problem 1 (Overlap Add and Save Methods)

i) `conv(x,h)` in MATLAB uses DFT in order to convolve x and h linearly. In fact, it uses `fft` to compute DFT of x and h and then multiplies their DFTs and finally takes the inverse DFT to obtain result (recall: convolution in time domain is equivalent to multiplication in frequency domain).

In this problem you are be familiar with overlap-add and overlap-save methods in order to compute linear convolution more easily than direct method when the length of input signal is large.

ii)

$$\begin{aligned} y[n] &= x[n] * h[n] \\ &= (\sum_{r=0}^{\infty} x_r[n - rB]) * h[n] \\ &= \sum_{r=0}^{\infty} (x_r[n - rB] * h[n]) \\ &= \sum_{r=0}^{\infty} y_r[n - rB]. \end{aligned}$$

iii) In order to compare elapsed time for running convolution operation in different methods, we should write down convolution formula without using `conv`. Since it is not so interesting, we just focus on theoretical results instead of time comparison.

iv) Assume that y_1 is the output of linear convolution between B-points signal x and P-points impulse response h (where $P < B$). So, length of y_1 is $B + P - 1$. We also know that the output of circular convolution of x and h has length B (it must have the same length as input signal x). If we call this signal y_2, we will have:

$$\begin{aligned} y_2[0] &= y_1[0] + y_1[B] \\ y_2[1] &= y_1[1] + y_1[B+1] \\ &\dots \\ y_2[P-2] &= y_1[P-2] + y_1[B+P-2] \\ &\text{and} \\ y_2[P-1] &= y_1[P-1] \\ &\dots \\ y_2[B-1] &= y_1[B-1] \end{aligned}$$

So, $y_2[n] = y_1[n]$ except for first $P - 1$ indices.

v)

$$x_r[n] = x[n + r(B - P + 1) - P + 1] \qquad 0 \le n \le B - 1$$
$$y[n] = \sum_{r=0}^{+\infty} y_r[n - r(B - P + 1) + P - 1]$$

where

$$\begin{cases} y_r[n] = y_{rp}[n] & P - 1 \le n \le B - 1 \\ y_r[n] = 0 & \text{otherwise} \end{cases}$$

and $y_{rp}[n]$ is the circular convolution of $x_r[n]$ with $h[n]$.

vi)

```
% overlap-save method: "save" in input(!)

% add P-1 extra zeros at input
extra zeros=zeros(1,P-1);
x=[extra zeros, x];
L=length(x);
y3=[];

tic
while(length(x)≥B)

% circular convolution using cconv; recall: cconv(x,h,length(x))
% returns the output of filter h when input is x.
% Lengths of input and output are equal to each other.

% circular convolve for one B-points window from x
temp3=cconv(x(1:B),h,B);

% save results without considering first P-1 points
y3=[y3, temp3(P:end)];

% shift window to the right considering P-1 points overlapped
% with preceding section.
x=x( ((B+1)-(P-1)) : end);

end
toc
```

Problem 2

i)

$$\begin{aligned} h[n] &= \int_{-1/2}^{1/2} H(e^{j2\pi f}) e^{j2\pi fn} df \\ &= \int_{-f_c}^{f_c} 1 \cdot e^{j2\pi fn} df = \frac{1}{j2\pi n} e^{j2\pi fn} \Big|_{-f_c}^{f_c} \\ &= \frac{\sin(2\pi f_c n)}{\pi n}. \end{aligned}$$

The ideal low pass filter is a non-causal infinite impulse response (IIR) filter.

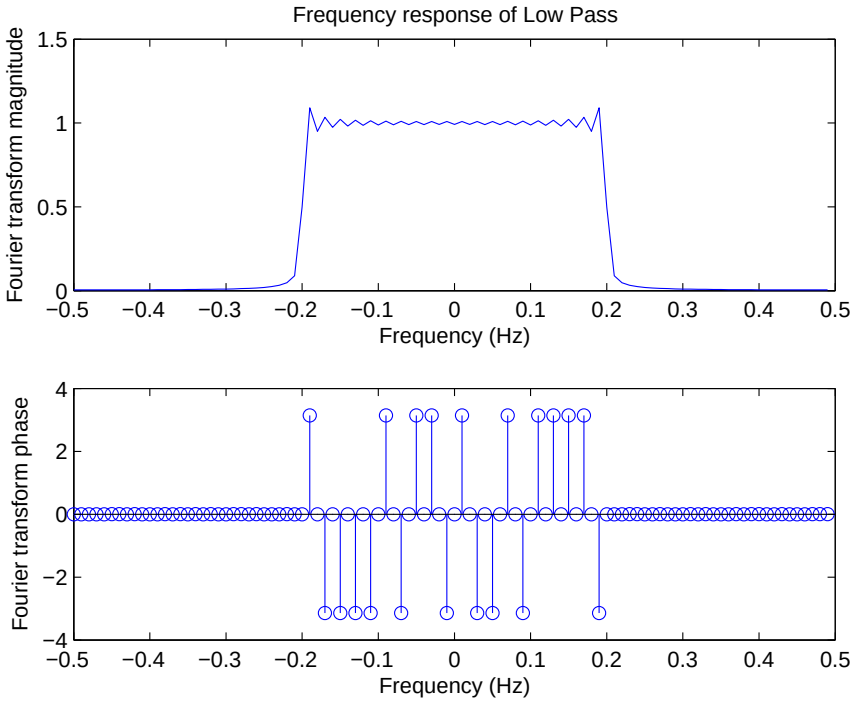

Figure 1: Plots of part 2.ii.

ii) It is not an ideal low pass. As mentioned before, the ideal filter has an infinite impulse response.
No, the reason is that the ideal LP filter is a non-causal filter while this LP filter is a causal and its impulse response starts at $n = 1$. It means that it is an ideal LP filter with an infinite $(N/2)$ delay.

iii) As mentioned in the previous part, we should remove the $N/2$ delay taken place by using $LowPass.m$ function. The following script is the corresponding code:

```
%Part iii:
clear all
N=100;

x0 = sin(0.05*[1:N])+sin(.1*[1:N])+sin(.2*[1:N])+ sin(1*[1:N]);
xlp=sin(0.05*[1:N])+sin(.1*[1:N])+sin(.2*[1:N]);
X0=fft(x0,N); %Fourier Transform of x0
plot((-N/2:N/2-1)/N,abs(fftshift(X0)));
hold on

H0=LowPass(.1,N);
plot((-N/2:N/2-1)/N,abs(fftshift(H0)),'r');
X1=X0.*H0;
plot((-N/2:N/2-1)/N,abs(fftshift(X1)),'g');
xlabel('Frequency (Hz)')
ylabel('Fourier transform magnitude')
x1=ifft(X1,N);
x1 = circshift(x1',N/2)'; % It is needed because the LowPass filter
% is started from 1 not from -N/2.
figure
stem(x0)
```

```
hold on
stem(x1,'r')
stem(xlp,'g')
```

In Fig. 2, the frequency response and time response of $x_0[n]$ and its filtered signal are depicted. In the frequency domain the blue curve is the frequency response of $x_0[n]$. The red curve is frequency response of the LP filter and finally the green curve is the frequency response of the filtered $x_0[n]$.

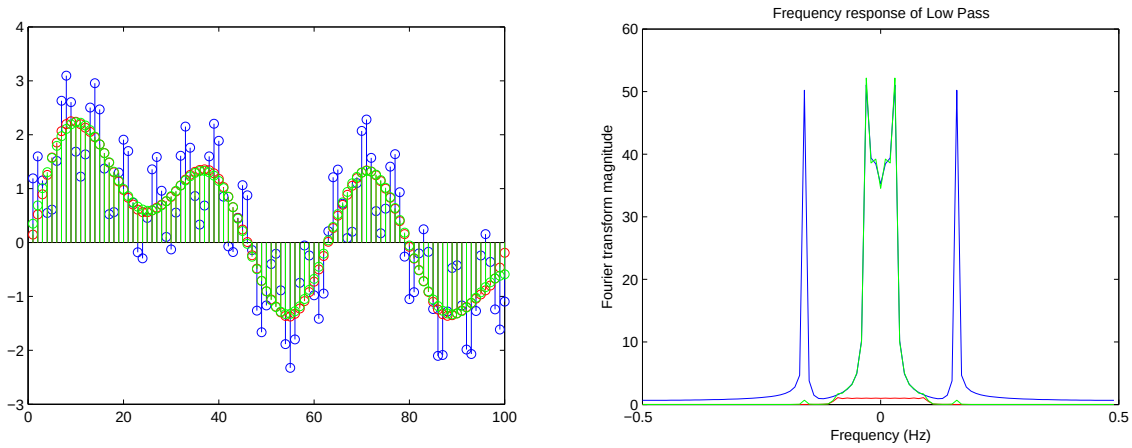

Figure 2: Plots of part 2.iii.

iv) By observing the spectrum of $x_u[n]$ and $x_{lp}[n]$ we can see that the spectrum of $x_u[n]$ contains the spectrum of $\frac{1}{L}x_{lp}[n]$ and its copies with interval $\frac{2\pi}{L}$ (or $\frac{1}{L}$ Hz). Therefore, if we filters $x_u[n]$ by a low pass filter with cut off frequency $\frac{1}{2L}$ Hz and gain L, the $x_{lp}[n]$ will be retrieved. In Fig. 3 the frequency response and the time response of the filtered signal are depicted. As we see, the resulted signal after decimation and interpolation is very close to the original signal.

```
%Part iv:
%Decimation:
M=2;
L=2;

figure
xd = x1(1:M:N); %down sample x1 by the factor M.
Nd=floor(N/M);
Xd = fftshift(fft(xd,Nd));
plot((-Nd/2:Nd/2-1)/(Nd),abs(Xd));

% Up sampling xd by the factor L.
%Zero Padding:
Nu=L*Nd;
xu=zeros(1,Nu);
xu(1:L:Nu)= xd;
Xu=fft(xu,Nu);
hold on
plot((-Nu/2:Nu/2-1)/(Nu),abs(fftshift(Xu)),'r');

%Interpolation:
```

```
% Using Ideal Low Pass Filter with Gain L and cut-off rate 1/2L:

LP=floor(Nu/(2*L));
H = L*LowPass(1/(2*L),Nu);
plot((-Nu/2:Nu/2-1)/(Nu),abs(fftshift(H)),'g');
xlabel('Frequency (Hz)')
ylabel('Fourier transform magnitude')
X4 = Xu.*H;
x4 = ifft(X4,Nu);

x4 = circshift(x4',Nu/2)';% It is needed because the LowPass filter
% is started from 1 not from -N/2.
figure
plot(x1(1:length(x4)));
hold on
plot(x4,'r');
```

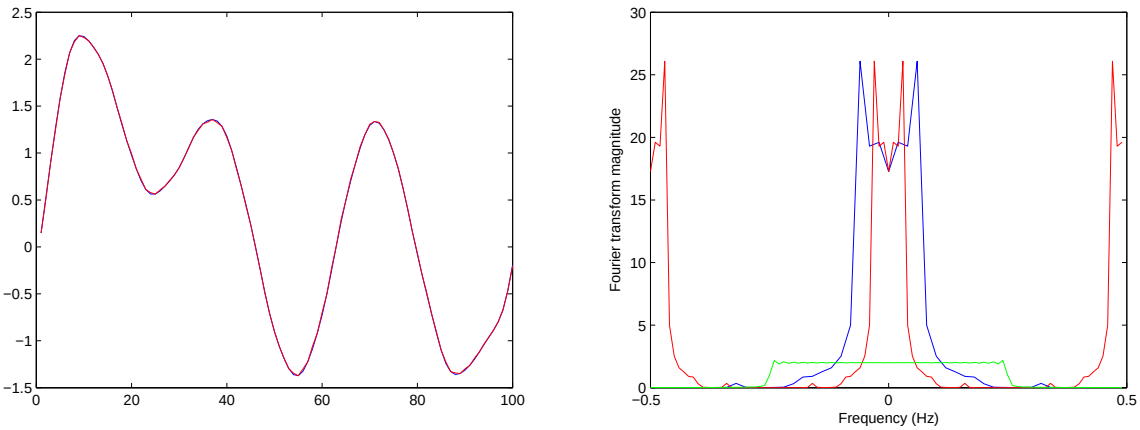

Figure 3: Plots of part 2.iv.

v) To retrieve the same signal, $M = L$. By performing down sampling the frequency response of the resulted signal is the frequency response of the original signal expanded in frequency domain by factor M. By carelessly expansion in frequency domain, the expanded frequency response of two consequence copies of the original frequency response, e.g. laid at 0 and 2π, may combine with each other and then the original signal can not be retrieved(We lose some information). Assume that $x_{lp}[n]$ has the bandwidth f_c. Thus, the bandwidth of the downsampled signal $x[n]$ is equal to Mf_c. If $Mf_c \geq \frac{1}{2}$, then the spectrum of the frequency response of $X_d(\frac{1}{2}) = X_{lp}(\frac{1}{2M}) + X_{lp}(1 - \frac{1}{2M})$. Therefore, by performing upsampling the frequency response of $x_u[n]$ changes from copies of frequency response of $x_{lp}[n]$ to copies of some distorted frequency response of $x_{lp}[n]$. This phenomena is called aliasing. In this problem, $M \leq 5$. In Fig. 4, 5, 6, the resulted signal for different values of M is depicted. As we can see, for $M = 6$, two signals are not the same any more.

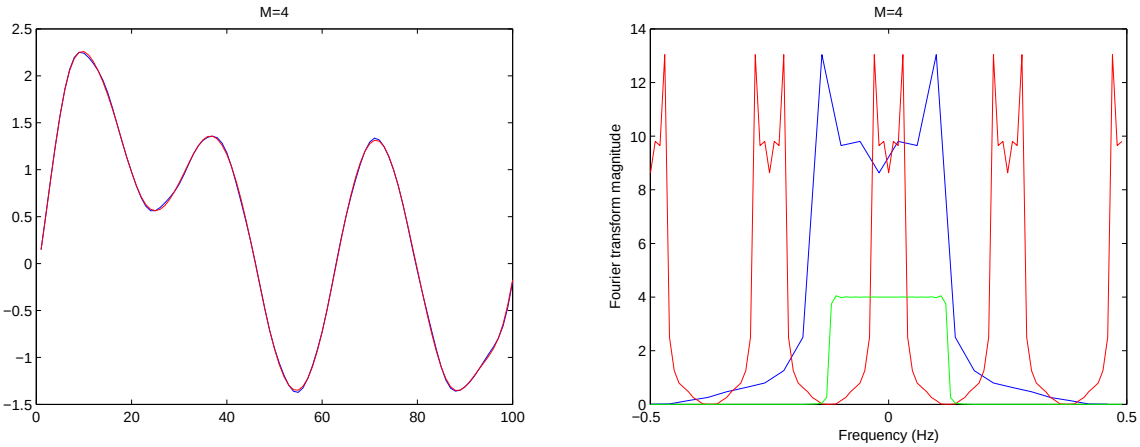

Figure 4: Plots of part 2.v where $M = 4$.

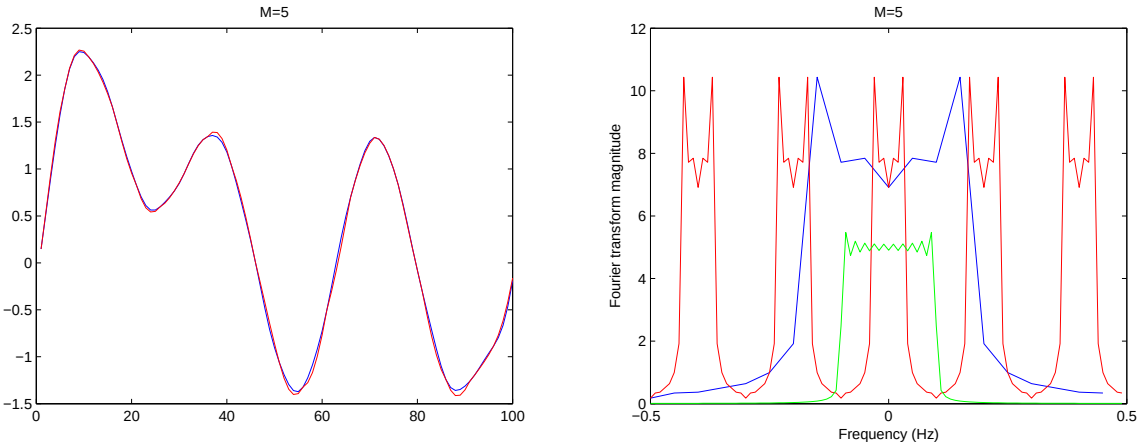

Figure 5: Plots of part 2.v where $M = 5$.

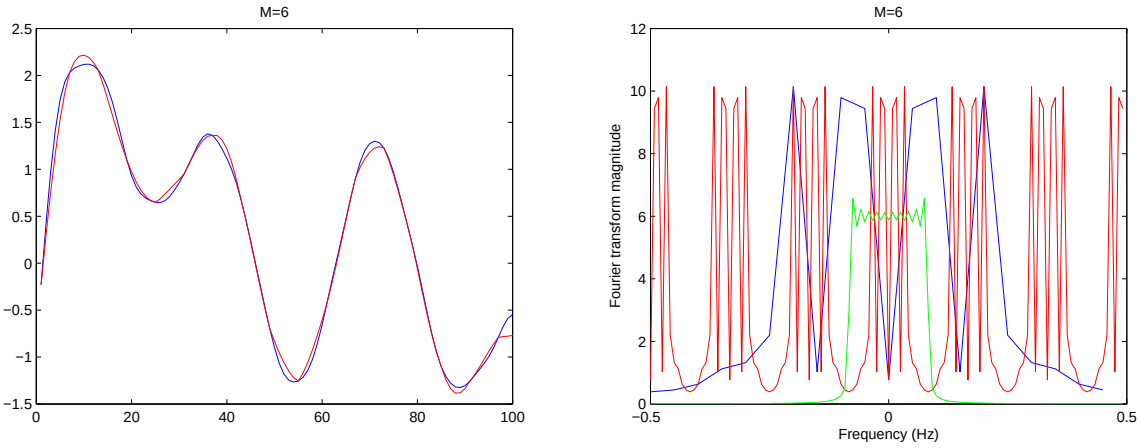

Figure 6: Plots of part 2.v where $M = 6$.