EE 542

Applications
In
Digital Signal Processing

Computer Assignment #1

Zexi Liu

Feb 6th, 2007

Electrical and Computer Engineering

Temple University

1. Introduction

1．1 Chebyshev Approximation

There are two types of *Chebyshev transfer functions*. In the *Type 1* approximation, the magnitude characteristic is equiripple in the passband and monotonic in the stopband, whereas in the Type 2 approximation, the magnitude response is monotonic in the passband and equiripple in the stop band.

The magnitude-squared response of the analog lowpass Type 1 Chebyshev filter $H_a(s)$ of Nth order is given by

$$\left|H_a(j\Omega)\right|^2 = \frac{1}{1+\varepsilon^2 T_N^2\left(\Omega/\Omega_p\right)},$$

where $T_N(\Omega)$ is the Chebyshev polynomial of order N:

$$T_N(\Omega) = \begin{cases} \cos\left(N\cos^{-1}\Omega\right), & |\Omega| \le 1 \\ \cosh\left(N\cosh^{-1}\Omega\right), & |\Omega| > 1 \end{cases}$$

The order N of the transfer function is determined from the attenuation specification in the stopband at a particular frequency. For example if at $\Omega = \Omega_s$, the magnitude is equal to 1/A, then from the above equations,

$$\left|H_a(j\Omega_s)\right|^2 = \frac{1}{1+\varepsilon^2 T_N^2\left(\Omega_s/\Omega_p\right)} = \frac{1}{1+\varepsilon^2\left\{\cosh\left[N\cosh^{-1}\left(\Omega_s/\Omega_p\right)\right]\right\}^2} = \frac{1}{A^2}.$$

Solving the above, we get

$$N = \frac{\cosh^{-1}\left(\sqrt{A^2-1}/\varepsilon\right)}{\cosh^{-1}\left(\Omega_s/\Omega_p\right)} = \frac{\cosh^{-1}\left(1/k_1\right)}{\cosh^{-1}\left(1/k\right)}$$

1．2 Gibbs Phenomenon

The causal FIR filters obtained by simply truncating the impulse response coefficients of the ideal filters exhibit an oscillatory behavior in their respective magnitude responses, which is more commonly referred to as the *Gibbs Phenomenon*.

The reason behind the Gibbs phenomenon can be explained by considering the truncation operation as multiplication by a finite-length window sequence $\omega[n]$ and by examining the windowing process in the frequency domain. Thus the FIR filter obtained by truncation can be alternatively expressed as

$$h_t[n] = h_d[n] \cdot \omega[n]$$

From the modulation theorem, the Fourier transform of the above equation is given by

$$H_t\left(e^{j\omega}\right) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d\left(e^{j\omega}\right)\Psi\left(e^{j(\omega-\varphi)}\right)d\varphi \, ,$$

where $H_t\left(e^{j\omega}\right)$ and $\Psi\left(e^{j\omega}\right)$ are the Fourier transforms of $h_t[n]$ and $\omega[n]$, respectively. The above equation implies that $H_t\left(e^{j\omega}\right)$ is obtained by a periodic continuous convolution of the desired frequency response $H_d\left(e^{j\omega}\right)$ with the Fourier transform $\Psi\left(e^{j\omega}\right)$ of the window.

2. Method

2.1 Impulse Invariance method

$$\omega_p = \frac{2\pi F_p}{F_T}$$

$$\omega_s = \frac{2\pi F_s}{F_T}$$

$$\Omega_p = T_s \cdot \omega_p$$

$$\Omega_s = T_s \cdot \omega_s$$

2.2 Bilinear transformation method

$$\omega_p = \frac{2\pi F_p}{F_T}$$

$$\omega_s = \frac{2\pi F_s}{F_T}$$

$$\Omega_p = \frac{2}{T_s} \cdot \tan\left(\omega_p\right)$$

$$\Omega_s = \frac{2}{T_s} \cdot \tan\left(\omega_s\right)$$

3. Matlab Program

3.1 Matlab Exercises M 9.5

```
% M9.5 page 521 (Textbook 3rd edition)
% ----------------------------------------------------------------
clc;                                % clear screen
clear;                              % clear workplace
%-----------------------------------------------------------------
% Parameter of the filter
%-----------------------------------------------------------------
FT = 100000;                        % Sampling Rate = 100 kHz
Fp = 10000;                         % Passband edge frequency = 10 kHz
Fs = 30000;                         % Stopband edge frequency = 30 kHz
ap = 0.4;                           % Peak passband ripple = 0.4 dB
as = -50;                           % minimum stopband attenuation = -50 dB
%-------------------------------------------------------------------

wp = (2 * pi * Fp) / FT;            % Normalized angular passband edge frequency
ws = (2 * pi * Fs) / FT;            % Normalized angular stopband edge frequency

%-----------------------------------------------------------------
% Impulse invariance method
%-----------------------------------------------------------------
TsI = 1;                                        % Let Ts = 1 for convenience
Wp = TsI * wp;                                  % The relationship between digital bandedge
Ws = TsI * ws;                                  % frequencies and analog bandedge frequencies
epsilon = sqrt((1 / ((10^(-ap / 20)^2))) - 1);  % epsilon = ... Eq. (4.43)
A = sqrt(1 / ((10^(as / 20))^2));               % A = ... Eq. (4.43)
k1 = 1 / sqrt((A^2 - 1) / epsilon^2);           % k1 = ... Eq. (4.43)
k = Wp / Ws;                                    % k = ... Eq. (4.43)
N = int8(0.5 + acosh(1 / k1) / acosh(1 / k));   % N = ... Eq. (4.43)
N = double(N);                                  % int8 -> double
[zerosI, polesI, gainI] = cheb1ap(N, ap);
[Num1I, Den1I] = zp2tf(zerosI, polesI, gainI);
[Num2I, Den2I] = lp2lp(Num1I, Den1I, Wp);
[Num3I, Den3I] = impinvar(Num2I, Den2I, 1 / TsI);
[FreqResponseI, FreqVectorI] = freqz(Num3I, Den3I, 512);
subplot(221);
plot(FreqVectorI/pi, 20 * log10(abs(FreqResponseI))); grid on;
axis([0 1 -60 5]);
xlabel('\omega/\pi');
```

```
ylabel('Gain (dB)');
title('Impulse Invariance Method');
subplot(222);
plot(FreqVectorI/pi, unwrap(angle(FreqResponseI))); grid on;
axis([0 1 -9 1]);
xlabel('\omega/\pi');
ylabel('Phase (radians)');
title('Impulse Invariance Method');

%-----------------------------------------------------------------
% Bilinear transformation method
%-----------------------------------------------------------------
TsB = 2;                                    % Let Ts = 2 for convenience
Wp = (2 / TsB) * tan(wp / 2);               % The relationship between digital bandedge
Ws = (2 / TsB) * tan(ws / 2);               % frequencies and analog bandedge frequencies
epsilon = sqrt((1 / ((10^(-ap / 20)^2))) - 1);   % epsilon = ... Eq. (4.43)
A = sqrt(1 / ((10^(as / 20))^2));           % A = ... Eq. (4.43)
k1 = 1 / sqrt((A^2 - 1) / epsilon^2);       % k1 = ... Eq. (4.43)
k = Wp / Ws;                                % k = ... Eq. (4.43)
N = int8(0.5 + acosh(1 / k1) / acosh(1 / k));   % N = ... Eq. (4.43)
N = double(N);                              % int8 -> double
[zerosB, polesB, gainB] = cheb1ap(N, ap);
[Num1B, Den1B] = zp2tf(zerosB, polesB, gainB);
[Num2B, Den2B] = lp2lp(Num1B, Den1B, Wp);
[Num3B, Den3B] = bilinear(Num2B, Den2B, 1 / TsB);
[FreqResponseB, FreqVectorB] = freqz(Num3B, Den3B, 512);
subplot(223);
plot(FreqVectorB/pi, 20*log10(abs(FreqResponseB))); grid on;
axis([0 1 -60 5]);
xlabel('\omega/\pi');
ylabel('Gain (dB)');
title('Bilinear Transformation Method');
subplot(224);
plot(FreqVectorB/pi, unwrap(angle(FreqResponseB))); grid on;
axis([0 1 -9 1]);
xlabel('\omega/\pi');
ylabel('Phase (radians)');
title('Bilinear Transformation Method');
```

3.2 Matlab Exercises M 10.2

```
% M10.2 page 584 (Textbook 3rd edition)
% ----------------------------------------------------------------
clc;                          % clear screen
clear;                        % clear workplace
% ----------------------------------------------------------------
wc1 = 0.1;                    % cut off frequency wc1= 0.1
wc2 = 0.9;                    % cut off frequency wc2= 0.9
% ----------------------------------------------------------------
% M = 5
% ----------------------------------------------------------------
M = 5;
n = -M : M;
num = wc1 * sinc(wc1 * n) - wc2 * sinc(wc2 * n);
[h,w] = freqz(num);
plot(w / pi, abs(h), 'r', 'LineWidth', 2); grid on; hold on;
axis([0 1 -0.2 1.4]);
% ----------------------------------------------------------------
% M = 10
% ----------------------------------------------------------------
M = 10;
n = -M : M;
num = wc1 * sinc(wc1 * n) - wc2 * sinc(wc2 * n);
[h,w] = freqz(num);
plot(w / pi, abs(h), 'g', 'LineWidth', 2); grid on; hold on;
axis([0 1 -0.2 1.4]);
% ----------------------------------------------------------------
% M = 50
% ----------------------------------------------------------------
M = 50;
n = -M:M;
num = wc1 * sinc(wc1 * n) - wc2 * sinc(wc2 * n);
[h,w] = freqz(num);
plot(w / pi, abs(h), 'LineWidth', 2); grid on;
axis([0 1 -0.2 1.4]);
xlabel('\omega/\pi');
ylabel('Magnitude');
title('Impulse Response');
legend('show');
legend('M = 5','M = 20');
```

4. Results

Fig. 1. Impulse Invariance Method

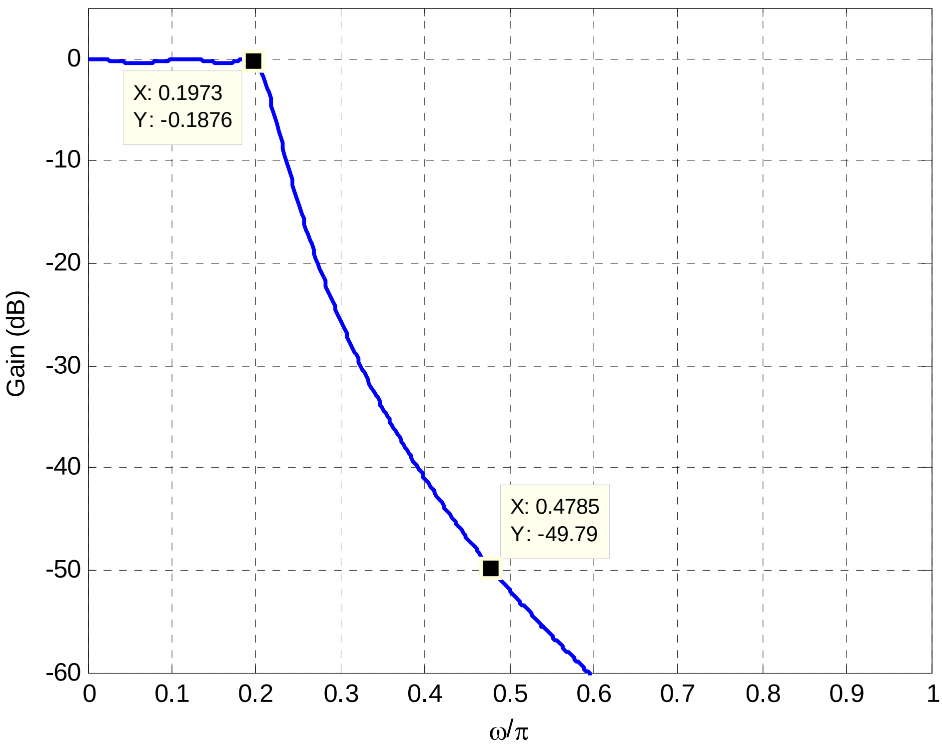


Fig. 2. Impulse Invariance Method

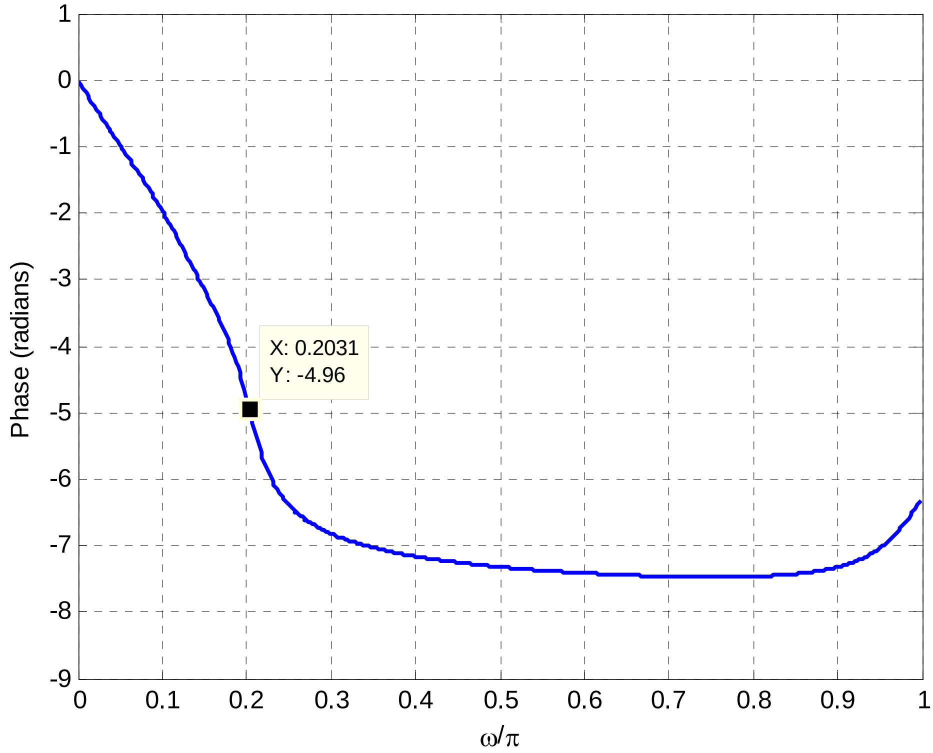


Fig. 3. Bilinear Transformation Method

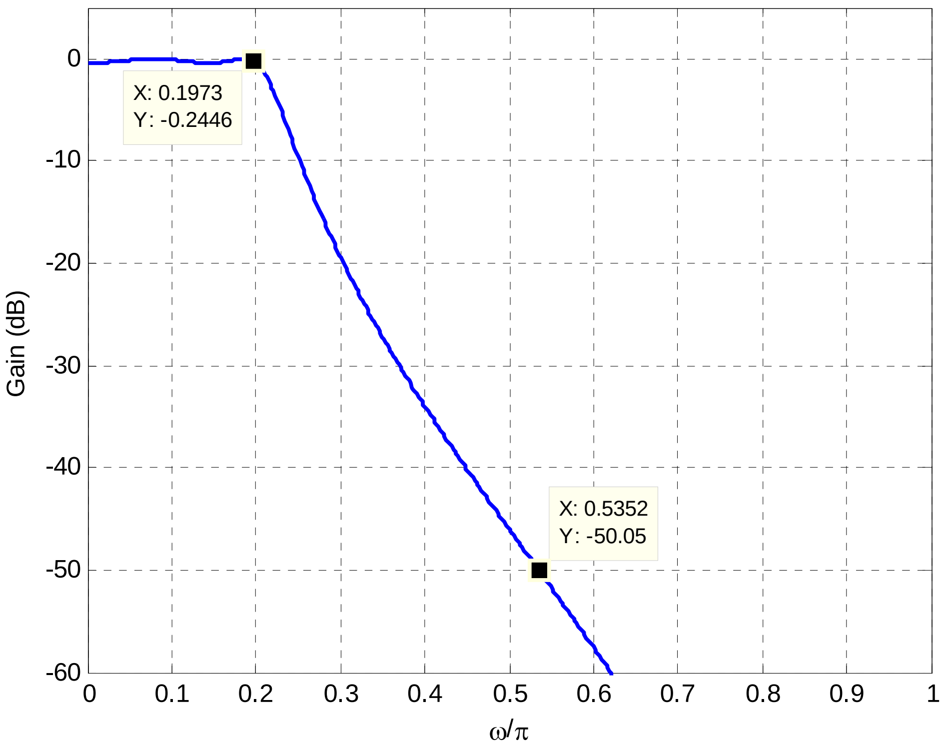
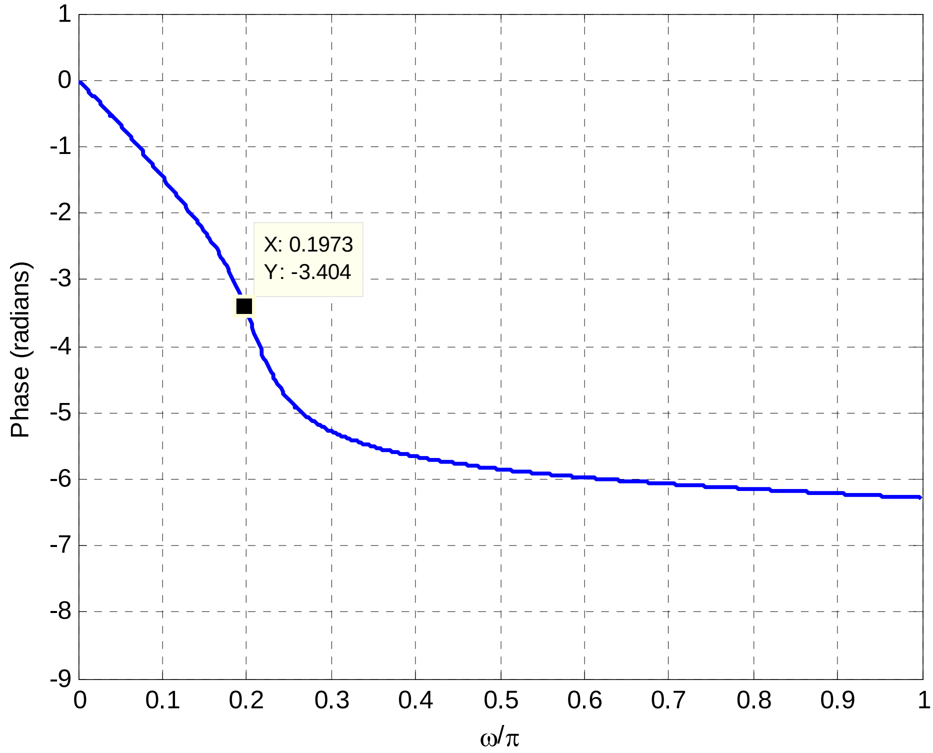


Fig. 4. Bilinear Transformation Method



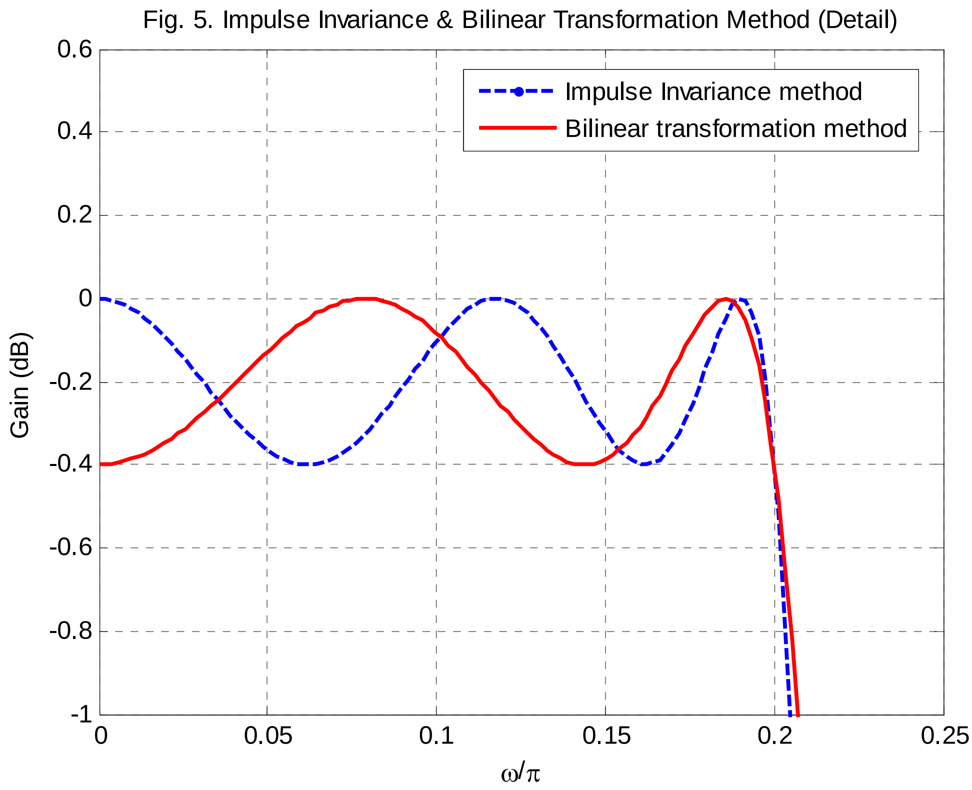


Fig. 5. Impulse Invariance & Bilinear Transformation Method (Detail)
Impulse Invariance method
Bilinear transformation method
Gain (dB)
ω/π
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.05
0.1
0.15
0.2
0.25

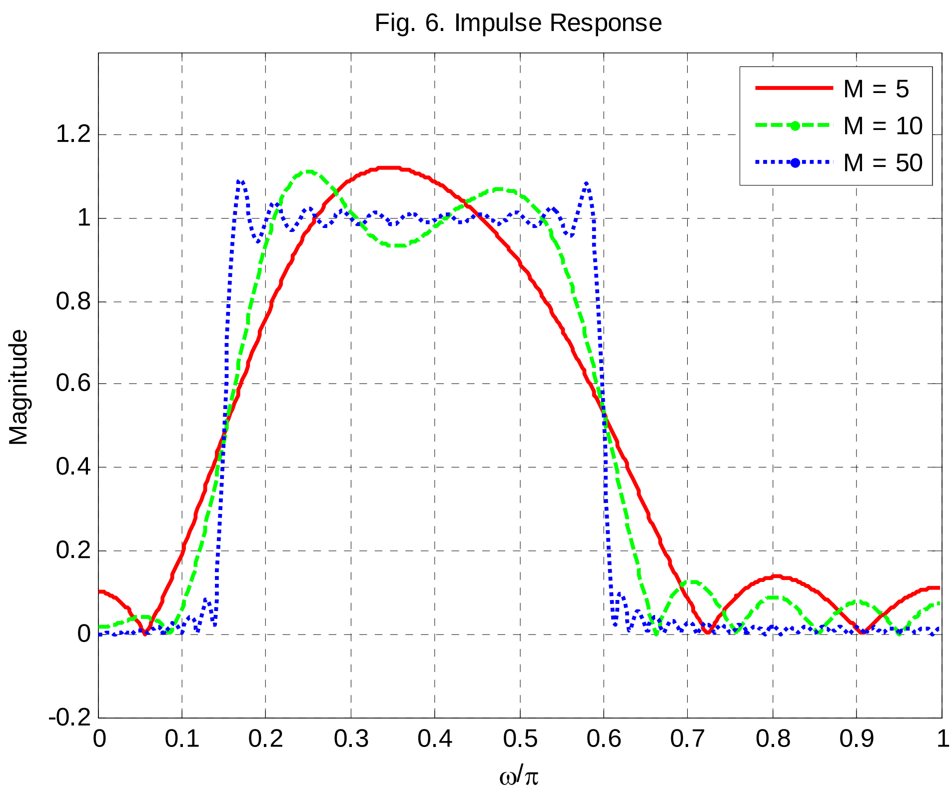
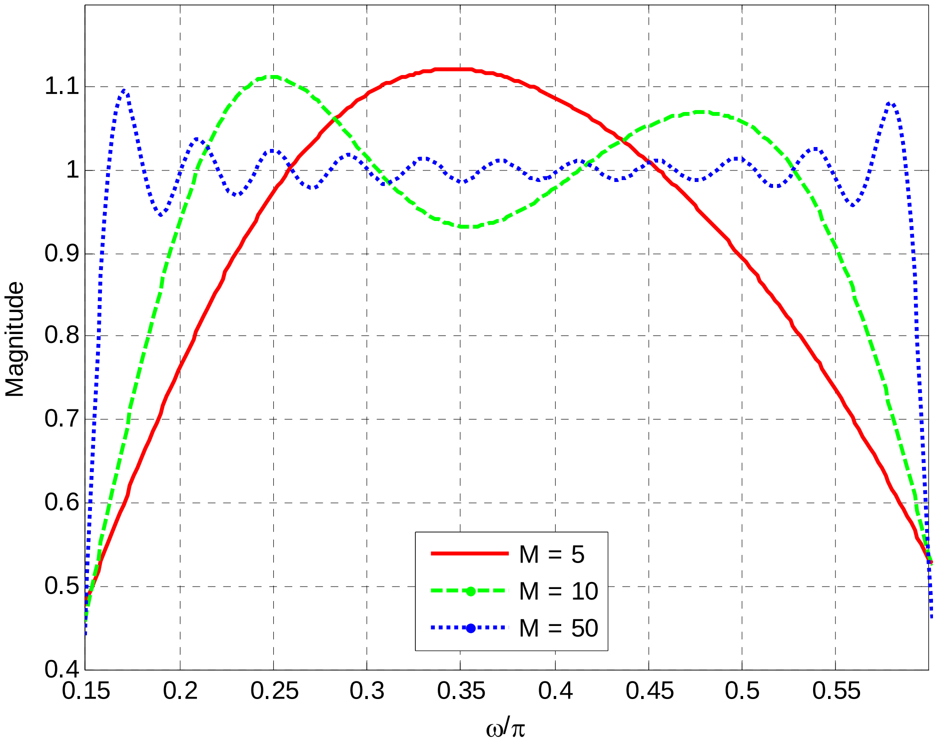


Fig. 6. Impulse Response
M = 5
M = 10
M = 50
Magnitude
ω/π
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Fig. 7. Impulse Response (Detail)



5. Conclusions

Comparing the order of two digital filters, the bilinear method is lower than the impulse invariance method. But the performance of impulse invariance method is slightly better than the bilinear method. At 30 Hz, the bilinear method has stopband attenuation of 46 db while the impulse invariance method has 52 dB. Their passband ripple is almost the same. And the impulse invariance method has a larger phase response after 10 kHz.

The oscillatory behavior of the magnitude response on both sides of the cutoff frequency is clearly visible. Moreover, as the length of the filter is increased, the number of ripples in both passband and stopband increases, with a corresponding decrease in the widths of the ripples. However, the heights of the rightmost and leftmost ripples are decreased much more slowly than other ripples.