Southern University of Science and Technology

Speech Signal Processing

Lab 10 Report

11510478 郭锦岳

Question 11.28

(a)

Code:

```
signal = audioread('s5.wav');

signal_8 = fxquant(signal, 8, 'round', 'sat');

signal_9 = fxquant(signal, 9, 'round', 'sat');

snr8 = snr(signal_8, signal);

snr9 = snr(signal_9, signal);

>>

snr8 = 37.8333

snr9 = 43.4176

snr9 – snr8 = 5.5843
```

The value roughly matches the expected amount 6 dB, as the bit number is increased by one.

(b)

To compute the values, I write a function called SNRplot, which plots two lines in a single run: the mu-law curve and the corresponding uniform curve.

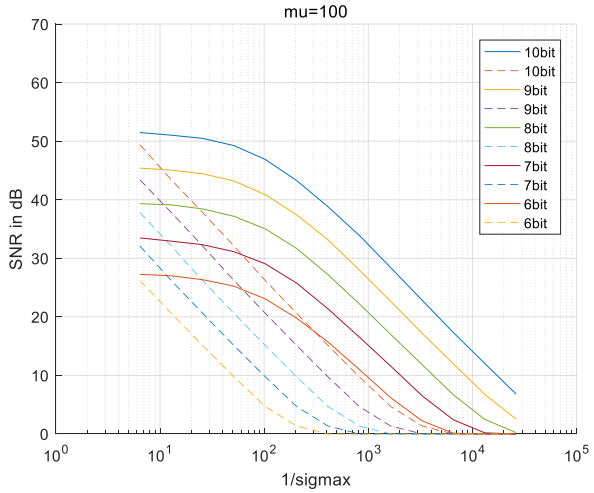
Code:

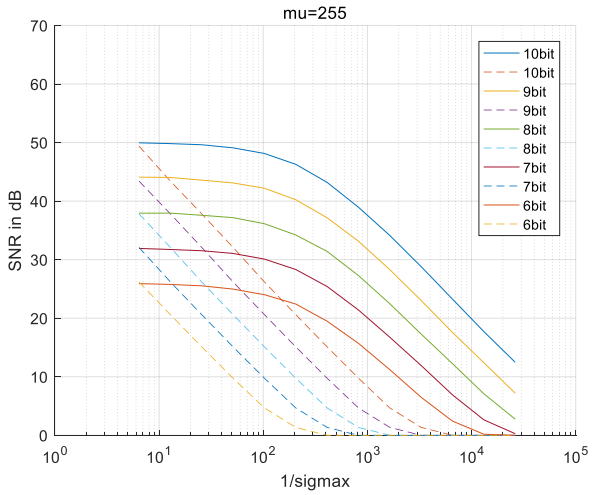
```
function [ signal ] = SNRplot( signal, mu, bit)
%SNRPLOR plot a single line
% signal: input signal array
% mu: factor for mu-law
% bit: quantization bit number

% x-axis: 1/sigma, 13 samples
% y-axis: SNR, 13 samples
```

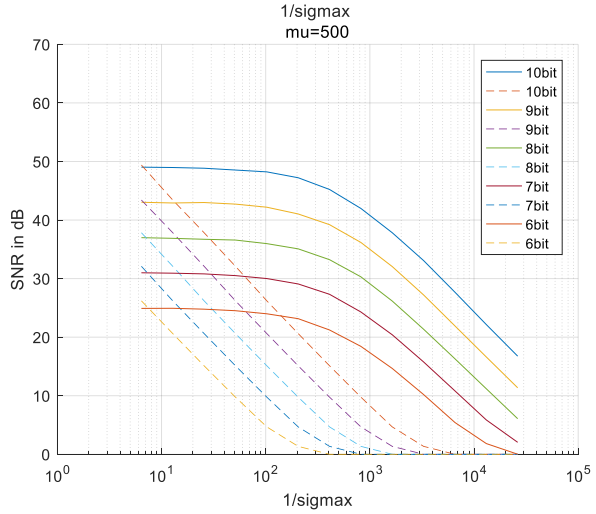
```
factor = 1*2.^(0:-1:-12)';
signal = factor*signal';
signal = signal';
SNR_mu = zeros(1,13);
SNR_uni = zeros(1,13);
standard = std(signal);
% mulaw and quantization
for i = 1:13
    signal_f = signal(:,i);
    signal_compress = mulaw(signal_f, mu);
    signal_q = fxquant(signal_compress, bit, 'round', 'sat');
    signal_expand = mulawinv(signal_q, mu);
    SNR_mu(i) = snr(signal_expand, signal_f);
    SNR_uni(i) = snr(fxquant(signal_f,bit,'round','sat'),signal_f);
end
semilogx(1./standard, SNR_mu);
semilogx(1./standard, SNR_uni,'--');
grid;
```

Running:

```
clear;clc
mu = 500;
signal = audioread('s5.wav');
hold on;
for i = 10:-1:6
    SNRplot(signal, mu, i);
end
hold off;
xlabel('1/sigmax');ylabel('SNR in dB');title(sprintf('mu=%d',mu));
legend('10bit','10bit','9bit','9bit','8bit','8bit','7bit','7bit','6bit','6bi
t');
```







As mu increases, the curves of mu-law quantizer could maintain as a flat line for a greater range. To reach the effect of 6bit mu-law quantizer, a uniform quantizer must have at least 9bit. The enhancement of mu-law quantizer is significant.

However, we can observe that at the beginning of the curves, SNR is slightly smaller when mu is bigger. I think this is because when variance is small, the quantization error might be amplified as mu gets bigger.

Question 11.30

To implement the ADPCM coders, I write two functions to realized the encoder and decoder.

Code:

```
function[ adpcm_y ]= adpcm_encoder(raw_y)

IndexTable = [-1, -1, -1, -1, 2, 4, 6, 8, -1, -1, -1, -1, 2, 4, 6, 8];

StepSizeTable = [7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 19, 21, 23, 25, 28, 31, 34, 37, 41, 45,
50, 55, 60, 66, 73, 80, 88, 97, 107, 118, 130, 143, 157, 173, 190, 209, 230, 253, 279, 307,
337, 371, 408, 449, 494, 544, 598, 658, 724, 796, 876, 963, 1060, 1166, 1282, 1411, 1552, 1707,
1878, 2066, 2272, 2499, 2749, 3024, 3327, 3660, 4026, 4428, 4871, 5358, 5894, 6484, 7132, 7845,
8630, 9493, 10442, 11487, 12635, 13899, 15289, 16818, 18500, 20350, 22385, 24623, 27086, 29794,
32767];

prevsample = 0;
previndex = 1;

Ns = length(raw_y);
n = 1;

raw_y = 32767 * raw_y;          % 16-bit operation

while (n <= Ns)
    predsample = prevsample;
    index = previndex;
    step = StepSizeTable(index);

    diff = raw_y(n) - predsample;
    if (diff >= 0)
        code = 0;
    else
        code = 8;
        diff = -diff;
    end

    tempstep = step;
    if (diff >= tempstep)
        code = bitor(code, 4);
        diff = diff - tempstep;
    end
    tempstep = bitshift(tempstep, -1);
    if (diff >= tempstep)
        code = bitor(code, 2);
        diff = diff - tempstep;
    end
    tempstep = bitshift(tempstep, -1);
    if (diff >= tempstep)
        code = bitor(code, 1);
    end
```

```
    diffq = bitshift(step, -3);
    if (bitand(code, 4))
        diffq = diffq + step;
    end
    if (bitand(code, 2))
        diffq = diffq + bitshift(step, -1);
    end
    if (bitand(code, 1))
        diffq = diffq + bitshift(step, -2);
    end

    if (bitand(code, 8))
        predsample = predsample - diffq;
    else
        predsample = predsample + diffq;
    end

    if (predsample > 32767)
        predsample = 32767;
    elseif (predsample < -32768)
        predsample = -32768;
    end

    index = index + IndexTable(code+1);

    if (index < 1)
        index = 1;
    end
    if (index > 89)
        index = 89;
    end

    prevsample = predsample;
    previndex = index;

    adpcm_y(n) = bitand(code, 15);
    n = n + 1;
end
```

Decoder:

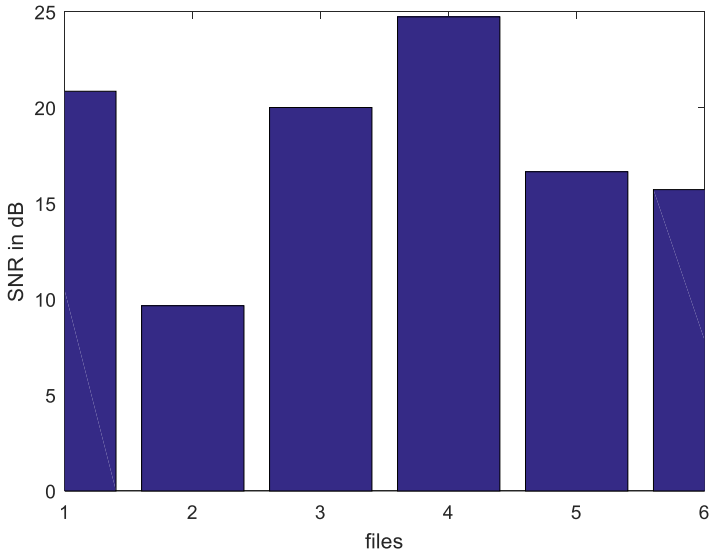
```
function [ decoded_y ] = adpcm_decoder( adpcm_y )

IndexTable = [-1, -1, -1, -1, 2, 4, 6, 8, -1, -1, -1, -1, 2, 4, 6, 8];

StepSizeTable = [7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 19, 21, 23, 25, 28, 31, 34, 37, 41, 45, 50, 55,
60, 66, 73, 80, 88, 97, 107, 118, 130, 143, 157, 173, 190, 209, 230, 253, 279, 307, 337, 371, 408,
449, 494, 544, 598, 658, 724, 796, 876, 963, 1060, 1166, 1282, 1411, 1552, 1707, 1878, 2066, 2272,
2499, 2749, 3024, 3327, 3660, 4026, 4428, 4871, 5358, 5894, 6484, 7132, 7845, 8630, 9493, 10442,
11487, 12635, 13899, 15289, 16818, 18500, 20350, 22385, 24623, 27086, 29794, 32767];

prevsample = 0;
previndex = 1;

Ns = length(adpcm_y);
n = 1;

while (n <= Ns)
    predsample = prevsample;
    index = previndex;
    step = StepSizeTable(index);
    code = adpcm_y(n);

    diffq = bitshift(step, -3);
    if (bitand(code, 4))
        diffq = diffq + step;
    end
    if (bitand(code, 2))
        diffq = diffq + bitshift(step, -1);
    end
    if (bitand(code, 1))
        diffq = diffq + bitshift(step, -2);
    end

    if (bitand(code, 8))
        predsample = predsample - diffq;
    else
        predsample = predsample + diffq;
    end

    if (predsample > 32767)
        predsample = 32767;
    elseif (predsample < -32768)
        predsample = -32768;
    end

    index = index + IndexTable(code+1);

    if (index < 1)
        index = 1;
    end
    if (index > 89)
        index = 89;
    end

    prevsample = predsample;
```

```
    previndex = index;

    decoded_y(n) = predsample / 32767;
    n = n + 1;
end

end
```

Running:

```
signal1 = audioread('s1.wav');
signal1_adpcm = adpcm_decoder(adpcm_encoder(signal1));
snr1 = snr(signal1_adpcm, signal1');
```

repeat for s1 to s6.



As for the result, the SNR varies for different input signals; Averagely the value is around 20dB. Even though the method I use is 16bit quantizing, the result does not seem to be satisfying. Maybe there is some problem of my code, and I need to work for further investigation.