SCD5 - Matlab - Übung 1

Daniel Klepatsch Thomas Reisinger

19. Oktober 2017

1. Übung
Quantisierung

1. Aufgabe *Fractional*

Vervollständigen Sie folgende Tabelle.

Wortlänge B	allgemein	8 Bits	16 Bits
kleinste positive Zahl (LSB)	2^(-(B-1))	0,0078125	0,000030517578125
größte positive Zahl (1-LSB)	1 - 2^(-(B-1))	0,9921875	0,999969482421875
betragsmäßig kleinste negative Zahl	0 - 2^(-(B-1))	-0,0078125	-0,000030517578125
Dynamik	(1 / (2^(-(B-1)))) - 1	127	32767
Präzision	2^(-B)	0,00390625	0,0000152587890625

Tabelle 1: Fractional-Format.

2. Aufgabe *Quantisierung von Fractional mit Runden*

Vervollständigen Sie folgende Tabelle.

x	x_Q binär	x_Q dezimal	x_Q hexadezimal
0.996	0111 1111	0.9921875	7Fh
0	0000 0000	0	00h
0.125	0001 0000	0,125	10h
-0.125	1111 0000	-0,125	F0h
0.004	0000 0001	0,0078125	01h
-1	1000 0000	-1	80h

Tabelle 2: Quantisierung im Fractional-Format mit $B = 8$ Bits.

3. Aufgabe *Zahlendarstellung IEEE 754-1985*

Geben sie Dynamik und Präzision einer Gleitkommazahl nach IEEE 754-1985 theoretisch (als Formel) und konkret für *double* und *single precision* an.
(Unter der Präzision versteht man in diesem Fall den aussteuerungsunabhängigen Quantisierungsfehler beim Runden aufgrund der Wortlängenbeschränkung der Mantisse.)

Bestimmen Sie die MATLAB Variablen "eps", "realmax" und "realmin" und machen Sie sich ihre Bedeutung klar.

4. Aufgabe *Quantisierungskennlinien*

Machen Sie sich mit der Funktion *quant2c.m* vertraut. Nehmen Sie die Quantisierungskennlinien zur Zweierkomplementdarstellung mit $B = 3$ Bit für Abschneiden bzw. Runden auf. Stellen Sie auch die Quantisierungsfehler dar. Vervollständigen Sie dazu das Programmgerüst *scd1_4_start.m*.

5. Aufgabe *Quantisierung eines Sinussignals*

Quantisieren Sie ein Sinussignal mit der Amplitude 1 mit den Quantisierungskennlinien aus der vorletzten Aufgabe für die Wortlänge von 4 Bits. Stellen Sie das Signal vor und nach der Quantisierung und den zugehörigen Quantisierungsfehler dar. (Erstellen sie auf dazu auf Basis von *scd1_4_start.m* ein neues Matlab Skript).

6. Aufgabe *SNR-Messungen*

an linearen Signalen, Sinussignalen und realen Sprachsignalen. Verwenden Sie hierfür das Programmgerüst *scd1_6_start.m*.

a) Messen Sie das Signal-Quantisierungsrausch-Verhältnis SNR für ein lineares Signal (gleichverteilt im Aussteuerbereich), ein sinusförmiges Signal und für das Sprachsignal "dsplab_speech.wav" in Abhängigkeit der Wortlänge (2 bis 16 Bits). Stellen Sie die Ergebnisse in dB in einem Bild dar. Vergleichen Sie die Ergebnisse des linearen Signals mit der Abschätzung $\text{SNR}_{dB} \approx B \cdot 6\text{dB}$.

b) Um wieviel ist das SNR beim sinusförmigen Signal besser als beim linearen Signal?

c) Um wieviel ist das SNR beim Sprachsignal schlechter als beim linearen Signal?

d) Verringern Sie die Amplitude der drei Signalformen um den Faktor 2 und wiederholen Sie die Messung. Um wieviel sinkt das SNR jeweils?

e) Stellen Sie beim Sinussignal (Sprachsignal) die Amplitude 1.02 ein (Übersteuerung) und wiederholen Sie die Messung. Um wieviel sinkt das SNR gegenüber Vollaussteuerung (in Abhängigkeit der Wortlänge)?

7. Aufgabe *Hörtests*

mit Sprachsignalen bei unterschiedlicher Auflösung des abgetasteten Signals.

a) Untersuchen Sie den Einfluss der Quantisierung auf Audiosignale durch "Hörtests" (z.B. mit dem Sprachbeispiel "dsplab_speech.wav"). Verwenden Sie dazu das Programm *scd1_7.m*. Bis zu welcher Mindestwortlänge ist das Sprachsignal noch verständlich?

a) In der Grundeinstellung des Programms ist das Signal so skaliert, dass keine Übersteuerung auftritt. Übersteuern Sie das Signal und wiederholen sie die Hörtests. Was stellen sie bei extrem starker Übersteuerung fest?

Inhaltsverzeichnis

1 3. Aufgabe **5**

2 4. Aufgabe **6**

3 5. Aufgabe **7**

4 6. Aufgabe **9**

5 7. Aufgabe **14**

1 3. Aufgabe

```
% #############  scd1_3.m: Dynamik und Praezision einer
   Gleitkommazahl #############
%

clear all;
close all;

% Erklaerung eps, realmax, realmin
% eps:      Abstand von einer angegebenen Zahl zur nächst höhren
% realmax:  groeßte endliche Gleitkommazahl
% realmin:  kleinste positive normalisierte Gleitkommazahl

% Formel fuer Berechnung der Gleitkommazahlen
% x = (-1)^S*2^(E-OS)*(1+Mf)
% S....VorzeichenBit (1 oder 0)
% E....Exponent single precision (1<=E<=254),
%               double precision (1<=E<=2046)
% OS...Offset Darstellung single precision (OS=127),
%                         double precision (OS=1023)
% Mf...Wert der Mantisse

% kleinste Positive Zahl
smallestSingleOwn = (-1)^0*2^(1-127)*(1+0);
smallestDoubleOwn = (-1)^0*2^(1-1023)*(1+0);

% groeßte Positive Zahl
biggestSingleOwn = (-1)^0*2^(254-127)*(1+(1-2^-23));
biggestDoubleOwn = (-1)^0*2^(2046-1023)*(1+(1-2^-52));

% kleinste und groeßte single- und double-precision Zahl
biggestDouble = realmax('double');
biggestSingle = realmax('single');
smallestDouble = realmin('double');
smallestSingle = realmin('single');

% Dynamik von single und double
dynDouble = biggestDouble/smallestDouble; % = 0.8079*10^-616
dynSingle = biggestSingle/smallestSingle; % = 2.8948*10^-76
% Matlab kann die Werte nicht darstellen, weil der Wert so groß
   ist

% Praezision von single und double (Abhäng von Exponent)
preDouble = 2^-23/2;
preSingle = 2^-52/2;
```

2 4. Aufgabe

```
% #############  scd1_4.m: Aufnahme von Quantisierungskennlinien
   #############
%

clear all
close all

% input signal
% interval from -1.5 to +1.5 in 0.02 steps
% range to exceeds -1 and +1 to demonstrate the saturation area
x = -1.5:0.02:1.5;

% Bit width
B = 3;

% quantization
xqr = quant2c(x,B,'r');  % rounding
xqf = quant2c(x,B,'f');  % truncation

% graphics
figure('Name','quantization ','NumberTitle','off');
subplot(2,2,1),plot(x,xqr,x,xqr,'.'),grid
xlabel('x \rightarrow')
ylabel('[x]_{Q,r}   \rightarrow')
title('rounding')
axis([-1.5,1.5,-1.2,1.2])
subplot(2,2,3),plot(x,xqr-x),grid
xlabel('x  \rightarrow')
ylabel('\Delta_r  \rightarrow')
subplot(2,2,2),plot(x,xqf,x,xqf,'.'),grid
xlabel('x  \rightarrow')
ylabel('[x]_{Q,f}   \rightarrow')
title('truncation')
axis([-1.5,1.5,-1.2,1.2])
subplot(2,2,4),plot(x,xqf-x),grid
xlabel('x  \rightarrow')
ylabel('\Delta_f  \rightarrow')
% end
```

3 5. Aufgabe

```
% #############  scd1_5.m: Aufnahme von Quantisierungskennlinien
   eines Sinus #############
%

clear all
close all

% input signal
t = 0:(pi/100):(2*pi);
sine = sin(t);

% Bit width
B = 4;

% quantization
sine_qr = quant2c(sine,B,'r');  % rounding
sine_qf = quant2c(sine,B,'f');  % truncation

% graphics
figure('Name','Quantization Sinus ','NumberTitle','off');

subplot(2,3,1),plot(t,sine),grid
xlabel('t\rightarrow')
ylabel('sine\rightarrow')
title('base signal')
axis([0,(2*pi),-1.5,1.5])

subplot(2,3,2),plot(t,sine_qr,t,sine_qr,'.'),grid
xlabel('t\rightarrow')
ylabel('sine_{Q,r}\rightarrow')
title('rounding')
axis([0,(2*pi),-1.5,1.5])

subplot(2,3,3),plot(t,sine_qf,t,sine_qf,'.'),grid
xlabel('t\rightarrow')
ylabel('sine_{Q,f}\rightarrow')
title('truncation')
axis([0,(2*pi),-1.5,1.5])

subplot(2,3,5),plot(t,sine_qr-sine),grid
xlabel('t\rightarrow')
ylabel('\Delta_r\rightarrow')
axis([0,(2*pi),-1,1])
axis 'auto y'

subplot(2,3,6),plot(t,sine_qf-sine),grid
```

```
xlabel('t\rightarrow')
ylabel('\Delta_f\rightarrow')
axis([0,(2*pi),-1,1])
axis 'auto y'

% end
```

4 6. Aufgabe

```
% ##################### scd1_6: SNR Messungen
   ########################

% measurement of signal-to-quantization noise power ratio

clear all
close all

% vector of wordlengths to be investigated
B = [2 3 4 5 6 7 8 9 10 11 12 13 14 15 16];

% linear input signal
x = -1:0.001:1;                          % input signal
S = sum(x.^2)/length(x);                 % signal power
SNR_l = zeros(size(B));
for k=1:length(B)
  xq = quant2c(x,k,'r');                 % quantized signal (
     rounding)
  N = sum((xq-x).^2)/length(x);          % noise power
  SNR_l(k) = S/N;                        % SNR
end


SNR_l_dB = 10*log10(SNR_l);
SNR_l_dB_Schaetzung = zeros(size(B));

for k=1:length(B)
    SNR_l_dB_Schaetzung(k) = B(k)*6;
end

SNR_Abweichung = SNR_l_dB - SNR_l_dB_Schaetzung;


% sinusoidal input signal
n = 0:0.001:1;                           % normalized time
x = sin(2*pi*n);                         % input signal
S = sum(x.^2)/length(x);                 % signal power
SNR_s = zeros(size(B));
for k=1:length(B)
  xq   = quant2c(x,k,'r');               % quantized signal (
     rounding)
  N = sum((xq-x).^2)/length(x);          % noise power
  SNR_s(k) = S/N;                        % SNR
end

% speech signal
```

```
[x,fs]=audioread('dsplab_speech.wav');
x = x/max(abs(x));                          % scaling
S = sum(x.^2)/length(x);                    % signal power
SNR_speech = zeros(size(B));
for k=1:length(B)-1
  xq = quant2c(x,k,'r');                    % quantized signal (
     rounding)
  N = sum((xq-x).^2)/length(x);             % noise power
  SNR_speech(k) = S/N;                      % SNR
end

% graphics
figure('Name','quantization SNR','NumberTitle','off');
subplot(2,1,1), plot(B,10*log10(SNR_l),'o',B,10*log10(SNR_s),'*',
   ...
  B,10*log10(SNR_speech),'+')
h = legend('linear','sinusiodal','speech');
xlabel('word length B [bits] \rightarrow')
ylabel('SNR [dB] \rightarrow')
hold on
plot(B,10*log10(SNR_l),':',B,10*log10(SNR_s),':',...
  B,10*log10(SNR_speech),':'),grid
hold off
subplot(2,1,2), plot(B,SNR_Abweichung), grid
xlabel('word length B [bits] \rightarrow')
ylabel('\Delta_{SNR} [dB] \rightarrow')
% end

% linear input signal
x = -1:0.001:1;                             % input signal
x = x/2;
S = sum(x.^2)/length(x);                    % signal power
SNR_l = zeros(size(B));
for k=1:length(B)
  xq = quant2c(x,k,'r');                    % quantized signal (
     rounding)
  N = sum((xq-x).^2)/length(x);             % noise power
  SNR_l(k) = S/N;                           % SNR
end

% sinusoidal input signal
n = 0:0.001:1;                              % normalized time
x = sin(2*pi*n);                            % input signal
x = x/2;
S = sum(x.^2)/length(x);                    % signal power
SNR_s = zeros(size(B));
for k=1:length(B)
  xq   = quant2c(x,k,'r');                  % quantized signal (
```

```
    rounding)
  N = sum((xq-x).^2)/length(x);          % noise power
  SNR_s(k) = S/N;                        % SNR
end

% speech signal
[x,fs]=audioread('dsplab_speech.wav');
x = x/max(abs(x));                       % scaling
x = x/2;
S = sum(x.^2)/length(x);                 % signal power
SNR_speech = zeros(size(B));
for k=1:length(B)-1
  xq = quant2c(x,k,'r');                 % quantized signal (
    rounding)
  N = sum((xq-x).^2)/length(x);          % noise power
  SNR_speech(k) = S/N;                   % SNR
end

% graphics
figure('Name','quantization SNR','NumberTitle','off');
plot(B,10*log10(SNR_l),'o',B,10*log10(SNR_s),'*',...
  B,10*log10(SNR_speech),'+')
h = legend('linear','sinusiodal','speech');
xlabel('word length B [bits] \rightarrow')
ylabel('SNR [dB] \rightarrow')
hold on
plot(B,10*log10(SNR_l),':',B,10*log10(SNR_s),':',...
  B,10*log10(SNR_speech),':'),grid
hold off
% end

% linear input signal
x = -1:0.001:1;                          % input signal
x = x/2;
S = sum(x.^2)/length(x);                 % signal power
SNR_l = zeros(size(B));
for k=1:length(B)
  xq = quant2c(x,k,'r');                 % quantized signal (
    rounding)
  N = sum((xq-x).^2)/length(x);          % noise power
  SNR_l(k) = S/N;                        % SNR
end

% sinusoidal input signal
n = 0:0.001:1;                           % normalized time
x = sin(2*pi*n);                         % input signal
x = x*1.02;
S = sum(x.^2)/length(x);                 % signal power
```

```
SNR_s = zeros(size(B));
for k=1:length(B)
  xq   = quant2c(x,k,'r');              % quantized signal (
     rounding)
  N = sum((xq-x).^2)/length(x);         % noise power
  SNR_s(k) = S/N;                       % SNR
end

% speech signal
[x,fs]=audioread('dsplab_speech.wav');
x = x/max(abs(x));                      % scaling
x = x*1.02;
S = sum(x.^2)/length(x);                % signal power
SNR_speech = zeros(size(B));
for k=1:length(B)-1
  xq = quant2c(x,k,'r');                % quantized signal (
     rounding)
  N = sum((xq-x).^2)/length(x);         % noise power
  SNR_speech(k) = S/N;                  % SNR
end

% graphics
figure('Name','quantization SNR','NumberTitle','off');
plot(B,10*log10(SNR_l),'o',B,10*log10(SNR_s),'*',...
  B,10*log10(SNR_speech),'+')
h = legend('linear','sinusiodal','speech');
xlabel('word length B [bits] \rightarrow')
ylabel('SNR [dB] \rightarrow')
hold on
plot(B,10*log10(SNR_l),':',B,10*log10(SNR_s),':',...
  B,10*log10(SNR_speech),':'),grid
hold off
% end

% a.) Bis 6 Bit stimmt die Abschätzung nicht, danach trifft sie
   zu.
%     Allerdings ist ein Offset von 6dB vorhanden.

% b.) Das SNR ist beim sinusförmigen Signal um ca 2dB besser als
   beim
% linearen Signal

% c.) das SNR ist beim Sprachsignal um ca. 10dB schlechter als
   beim
% linearen Signal

% d.) das SNR sinkt bei ab einer Wortlänge von 4Bits um ca. 7dB.
```

```
% e.) beim Sinussignal ist ab einer Wortlänge von ca. 8Bits eine
   Krümmung
% des Grafen zu sehen. Beim Sprachsignal ist ab einer Wortlänge
   von ca.
% 13Bits eine Krümmung des Grafen zu sehen.
```

5 7. Aufgabe

```
% ################################ scd1_7: Hörtests
   ##################################

clear all
close all

% load audio signal
fprintf('SCD1_7 : Effects of quantization\n')
% filename = input('name of audio file (*.wav): ','s');
[x,fs]=audioread('dsplab_speech.wav');
Ts = 1/fs;
x = x/max(abs(x));   % scaling
x = x*200;
fprintf('original signal) \n')
soundsc(x,fs);
pause(1)
B = input('word length (B=8): ');
if isempty(B)
  B = 8;
end
x = (1-2^(-(B-1)))*x;
L = length(x);
S = sum(x.^2)/L;          % signal power (energy)
xq = quant2c(x,B,'r');  % quantized signal (rounding)
N = sum((xq-x).^2)/L;    % noise power (energy)
SNR = S/N;
fprintf('quantization SNR = %g dB\n',10*log10(SNR));
FIG1=figure('Name','quantized audio signal','NumberTitle','off');
t=0:Ts:(L-1)*Ts;
plot(t,x,t,xq,'.')
grid, xlabel('t [s] \rightarrow'), ylabel('[x]_Q(t) \rightarrow')
fprintf('quantized signal %g bits \n\n',B)
soundsc(xq,fs);
% end

% a.) das Sprachsignal ist bis zu einer Wortlänge von 4 Bit
   verständlich.

% b.) bei Extremer Übersteuerung ist die Sprachausgabe auch noch
   mit 2 Bit
% erkennbar, jedoch ist die Ausgabe lauter.
```