Laborationsrapport i TSKS10 *Signaler, Information och Kommunikation*

Alexander Yngve
aleyn573, 930320-6651

8 maj, 2015

1 Inledning

Målet med den här laborationen var att demodulera en smalbandig I/Q-modulerad signal skickad från en tänkt radiostation och lyssna på dess innehåll. Signalens innehåll består av musikaliska melodier, ordspråk och vitt brus. En del av uppgiften var att identifiera ordspråken. Förutom demodulering skulle även signalens bärfrekvens bestämmas samt ekoeffekter tas hänsyn till. Resultatet av laborationen blev kända värden på signalens bärfrekvens, ekotidsfördröjning samt en användbar signal där ordspråken kunde höras.

Från labbhandledningen fås följande information om signalen:

- Radiostationen sänder ut signalen $x(t) = x_I(t)cos(2\pi f_c t) - x_Q(t)sin(2\pi f_c t) + z(t)$, där f_c är signalens bärfrekvens och $z(t)$ är andra signaler ämnade åt någon annan. x_I och x_Q är de intressanta signalerna med relevant innehåll.
- På grund av ekoeffekter i radioutbredningsmiljön så tar vi emot signalen $y(t) = x(t-\tau_1) + 0.9x(t-\tau_2)$.
- Bärfrekvensen f_c är en multipel av 19 kHz.
- Den mottagna signalen lågpassfiltreras och samplas sedan med en frekvens på $f_s = 400000$ Hz.

2 Metod

Laborationen kan delas in i tre deluppgifter, bestämming av bärfrekvens samt filtrering i smalbandet, bestämning av ekots tidsfördröjning samt filtrering av detta och slutligen I/Q-demodulering. MATLAB användes som verktyg för att behandla signalen.

2.1 Bärfrekvens och filtrering

Bärfrekvensen f_c fås med hjälp av fouriertransformen $Y(f)$ till signalen $y(t)$. Figur 1 visar amplitudspektrumet $|Y(f)|$. Ur figuren fås att det finns innehåll i närheten av tre olika bärfrekvenser. De frekvenser som matchar kritieriet att vara en multipel av 19 kHz är våra möjliga bärfrekvenser.

- Signal y_1 med $f_{c1} = 38$ kHz
- Signal y_2 med $f_{c2} = 76$ kHz
- Signal y_3 med $f_{c3} = 114$ kHz

Signalerna på de olika frekvenserna filtreras ut med hjälp av ett bandpassfilter med övre respektive undre gräns $f_{ci} - B/2$ och $f_{ci} + B/2$ där $i = 1,2,3$. Då vi vet att signalen ska innehålla hörbart ljud väljs bandbredden $B = 20000$ Hz.

Figur 2 visar de filtrerade smalbandssignalerna i tidsdomänen. Signalerna y_1 och y_2 med bärfrekvens f_{c1} och f_{c2} ser ut att vara rent brus eller snarlikt. Signalen y_3 med bärfrekvens f_{c3} ser däremot ut att ha korrekt innehåll, två delar som skulle kunna vara toner och tal och en avslutande del brus.

2.2 Hantering av eko

För att få bort ekoeffekterna i signalen måste vi först veta ekots tidsfördröjning. Detta görs genom en autokorrelation på signalen. Signalen y_2 som bara innehåller brus används då denna typ av vågform ger ett tydligare resultat enligt kursboken. I figur 3 syns huvudtoppen vid tiden $t = 0$ och sidotoppen vid $t = 0.430$, alltså är ekots tidsfördröjning $\tau = \tau_1 - \tau_2 = 0.430$ sekunder.

Med en känd tidsfördröjning samt amplitud på ekot från avsnitt 1 kan nu den ekofria signalen y_3' tas fram från y_3. Detta görs genom $y_3'(t) = y_3(t) - 0.9y_3'(t-\tau)$ för alla $t > \tau$. Då det är uppenbart att det inte finns något eko för $0 \leq t \leq \tau$ gäller $y_3'(t) = y_3(t)$ i det intervallet.

Implementationen i MATLAB utför ekodämpningen på ett helt block av sampel i taget istället för en och en. Detta för att få en snabbare exekveringstid, i övrigt är det samma algoritm. Blockets storlek är antalet sampel som motsvarar $\tau = 0.430$ sekunder, det vill säga $f_s\tau = 172000$ sampel.

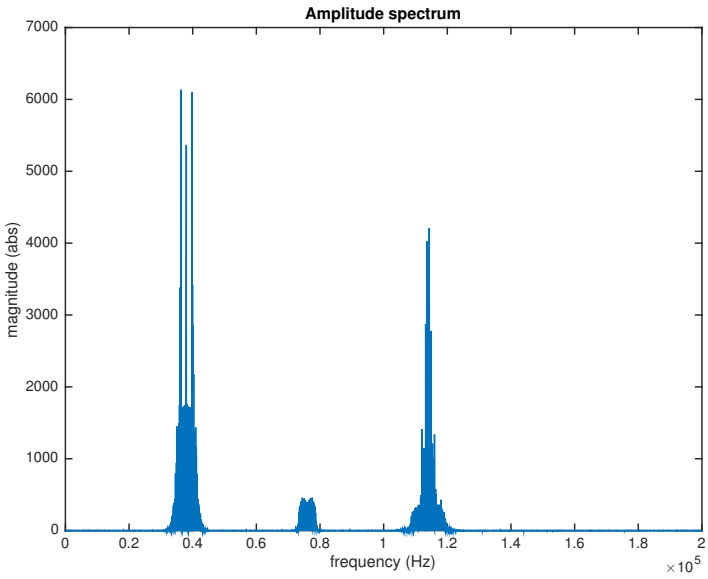
2.3 I/Q-demodulering

Nu återstår endast demodulering innan signalen kan spelas upp som hörbart ljud. Från kursboken har vi givet $x_I(t) = \mathscr{H}^{LP}_{B/2}\{2x(t)cos(2\pi f_c t)\}$ och $x_Q(t) = -\mathscr{H}^{LP}_{B/2}\{2x(t)sin(2\pi f_c t)\}$, där i vårt fall $x(t) = y_3'(t)$ enligt avsnitt 2.2, $f_c = f_{c3} = 114000$ Hz och $B = 20000$ Hz enligt avsnitt 2.1. Basbandssignalerna x_I och x_Q innehåller nu hörbart ljud som kan spelas upp i valfri mediaspelare!

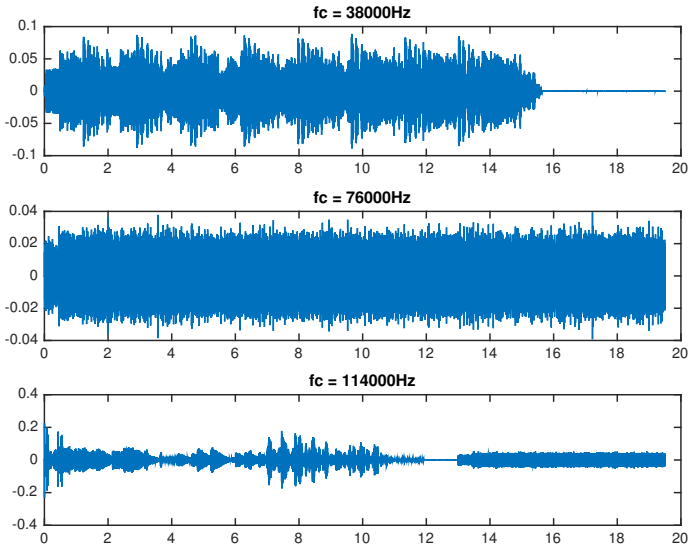
3 Resultat

Den sökta informationen är:

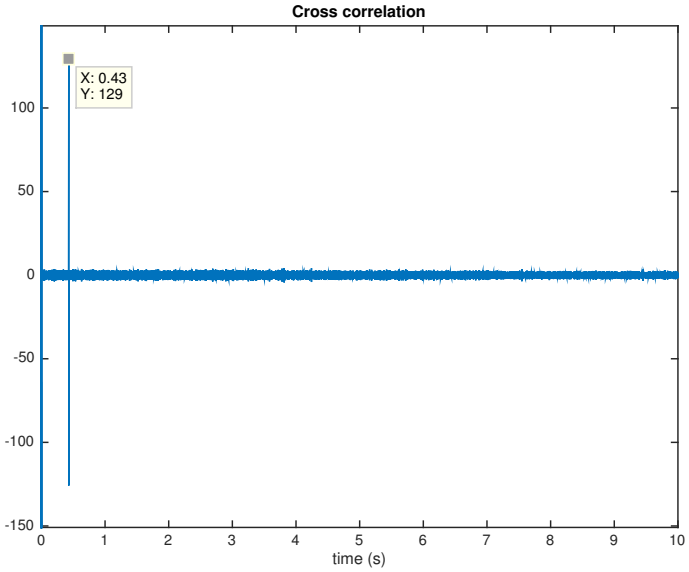
- Bärfrekvensen för nyttosignalen är $f_c = 114000$ Hz.
- Differensen $\tau_2 - \tau_1 = 0.430$ s.
- Ordspråket i I-signalen är ”även den mest skröpliga mussla kan innehålla en pärla”.
- Ordspråket i Q-signalen är ”skrattar bäst som skrattar sist”.



Figur 1: Amplitudspektrum



Figur 2: Radiostationens tre signaler i tidsdomänen



Figur 3: Autokorrelation för att bestämma ekots tidsfördröjning

A Programkod

```
clc;clear;close all;

% Read signal
[y, fs, b] = wavread('signal-aleyn573.wav');
L = length(y);

% Transform
f_axis = fs/2*linspace(0, 1, L/2);
Y = fft(y);

% Check carrier frequencies
plot(f_axis, abs(Y(1:L/2)));
title('Amplitude spectrum');
xlabel('frequency (Hz)');
ylabel('magnitude (abs)');
pause;

% Check which signal contains relevant data
bw = 20000; % Bandwidth
fc1 = 38000;
fc2 = 76000;
fc3 = 114000;
t_axis = linspace(0, 19.5, L)';

[B, A] = butter(10, [fc1 - bw/2, fc1 + bw/2]/(fs
    /2));
y1 = filter(B, A, y);
[B, A] = butter(10, [fc2 - bw/2, fc2 + bw/2]/(fs
    /2));
y2 = filter(B, A, y);
[B, A] = butter(10, [fc3 - bw/2, fc3 + bw/2]/(fs
    /2));
y3 = filter(B, A, y);

subplot(3,1,1);
plot(t_axis, y1);
title(['fc = ' num2str(fc1) 'Hz']);
subplot(3,1,2);
plot(t_axis, y2);
title(['fc = ' num2str(fc2) 'Hz']);
subplot(3,1,3);
plot(t_axis, y3);
title(['fc = ' num2str(fc3) 'Hz']);
pause;

% y3 (114 kHz) seems to be the right one

% Cross-correlation of white noise (y2) to find
    echo time delay
[corr, lags] = xcorr(y2);
corr = corr(lags > 0); % Plot only positive time
lags = lags(lags > 0);
subplot(1,1,1);
plot(lags/fs, corr);
xlabel('time (s)');
title('Cross correlation');
pause;

tau = 0.43; % Difference in seconds from xcorr
    plot
diff = tau*fs; % Difference in samples

% Echo cancellation
y_echo_fix = zeros(size(y3));
y_echo_fix(1:diff) = y3(1:diff);

for i=1:42
    y_echo_fix(i*diff+1:(i+1)*diff) = y3(i*diff
        +1:(i+1)*diff) - 0.9*y_echo_fix((i-1)*
        diff+1:i*diff);
end

% I/Q-demodulation
[B, A] = butter(10, bw/(fs/2), 'low');

i_carrier = cos(2*pi*fc3*t_axis);
q_carrier = sin(2*pi*fc3*t_axis);

y_i = filter(B, A, 2*y_echo_fix.*i_carrier);
y_q = -filter(B, A, 2*y_echo_fix.*q_carrier);

% Playback
i = decimate(y_i, 4);
q = decimate(y_q, 4);

%soundsc(i, fs/4);
%soundsc(q, fs/4);
```