0.1 Introduction

I denne del af opgaven vil jeg vil jeg nedsamples signalet s1 med en faktor 4, hvorefter jeg vil sammenligne den med originalen ved bl.a plots og lyd.

0.2 Fremgangsmåde

Listing 1: Matlab kode for øvelse 5 & 6

```
%Definer nedsampling af signal
y(5).samples = downsamples(y(1).samples, 4);
y(5).name = "Resampling of " + y(1).name;

y(5).samples = y(5).samples';
% Antal af samples
y(5).nS = length(y(5).samples);
%Udregner x aksen for signalet
y(5).time = [0:y(5).nS-1]*(1/Fs);

figure
subplot(1,2,1)
plot(y(1).time, y(1).samples)
title(y(1).name)
xlabel("Time(s)")
ylabel("Amplitude(~)")
subplot(1,2,2)
plot(y(5).time, y(5).samples)
title(y(5).name)
xlabel("Time(s)")
ylabel("Amplitude(~)")

soundsc(y(5).samples, Fs/4);
```

0.3 Resultater

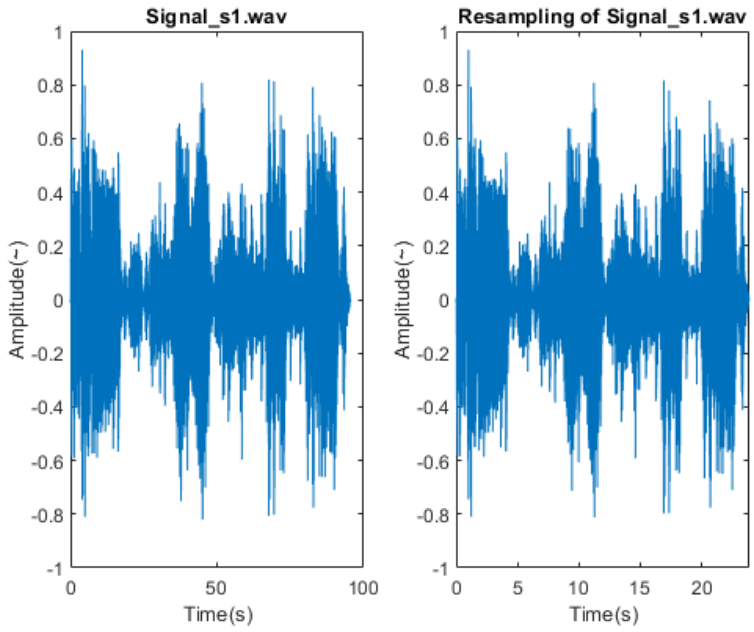

Figure 1: Max-værdier for de fire mono-signaler

0.4 Diskussion

Jeg benytter Matlab funktionen *downsamples*() til at fjerne hvert fjerde element i mit samples-array. Herefter korrigere der for nedsampling på frekvensen og samplestiden. Signalet før og efter plottes så som subplots, så de bedre kan sammenlignes, som ses på figur 7. Der kan ikke ses den store forskel, da samplingsraten stadig er relativt høj.

Derfor benytter jeg også *soundsc*() funktionen i Matlab til at lytte til begge signaler, hvor jeg kan hører en klar forskel, med lavere kvalitet lyd efter nedsamplingen. Det bemærkes her at frekvensen skal justeres for at lyden spiller i den rigtige hastighed, da der jo er fire gange mindre samples i signalet.

0.5 Konklusion

Jeg konkludere at en nedsampling af et signal vil forværre kvaliteten af signalet, da det ikke kan gengive det 'originale' signal lige så præcist. Desuden finder jeg også at en nedsamling også vil førre til et krav om justering af frekvensen.