Assignment 1

Signal Processing and Aircraft Flyover

AE4463P-23: Advanced Aircraft Noise

E. Oosthoek, J. Bogaert

Delft University of Technology

TU Delft

Assignment 1

Signal Processing and Aircraft Flyover

by

E. Oosthoek, J. Bogaert

Student Name	Student Number
Elisabeth Oosthoek	5056470
Joshua Bogaert	5298601

Instructors: Prof. dr. ir. M. Snellen, Prof. dr. D.G. Simons
Faculty: Faculty of Aerospace Engineering, Delft

Cover: https://unsplash.com/photos/white-and-blue-airplane-under-white-clouds-during-daytime-bNVbyBl870A
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

1
Exercise 2.7

1.1. Part a

The sampling theorem dictates that the sampling frequency should be F $\geq$ 2B. However the LPF introduces a deterioration of the signal of 1 kHz starting at the upper bound of the cutoff frequency, f_c.

Hence, in order to guarantee that aliasing does not occur, F is chosen larger than 2B (where B = 10 kHz, is the bandwidth of the speech signal)and is equal to F = 22 kHz. This makes sure that the full bandwidth can be reconstructed before the cutoff frequency. Furthermore, f_c is chosen to be 10 kHz (11-1 due to filter specifications). Below a sketch can be found of the above mentioned filter applied together with the the speech signal.

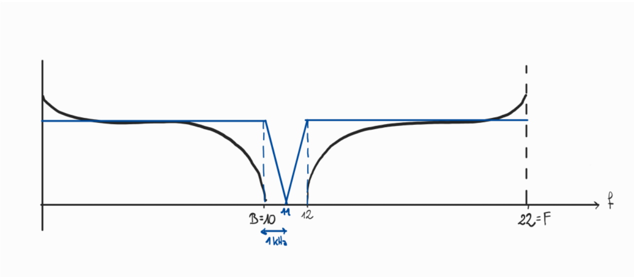


Figure 1.1: Sketch of the above mentioned situation

1.2. Part b

Due to F = 15 kHz aliasing occurs because 15 $\leq$ 2B. The repetition of the signal starts at 5 kHz as can be seen on Figure 1.2 . In case of an AAF the cut-off frequency should be: f_c = F/2. This eliminates all frequencies larger than F/2 and avoids alias for f_c = F/2. However, one has to account for the 1 kHz deterioration of the signal and thus as result the cut of frequency is 6.5 kHz

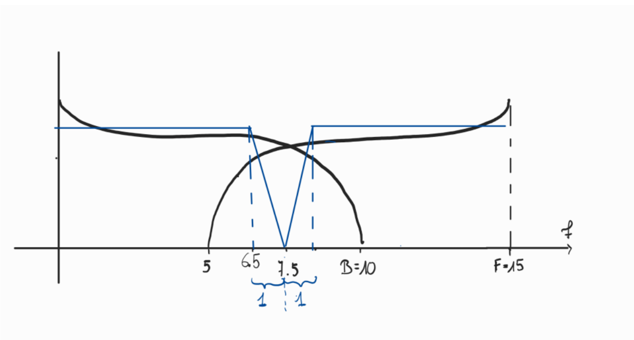


Figure 1.2: Sketch of the above mentioned situation

2
Aircraft Flyover

Within this chapter theory explained within the introduction lecture and the reader is utilised extensively [2][4]. Additionally [3] was utilised in support to the main course material. The code developed, can be found in chapter 3.

2.1. Question 1

To analyse the aircraft flyover, pressure measurements were taken utilising a microphone. These were sampled at a rate of 40 kHz (sample frequency). The data set itself provided pressure measurements, expressed in pascal for full duration of the flyover (leading to 730000 data points). Using the amount of data points and the sample frequency, the duration of the flyover could be determined to be 18.25 s. The following computation was utilised to determine duration of the signal: $\frac{1}{40\,\text{kHz}}(730000-1) = 18.25$ s. Utilising this end-time and the frequency of measurements the pressure data could be visualised in Figure 2.1.

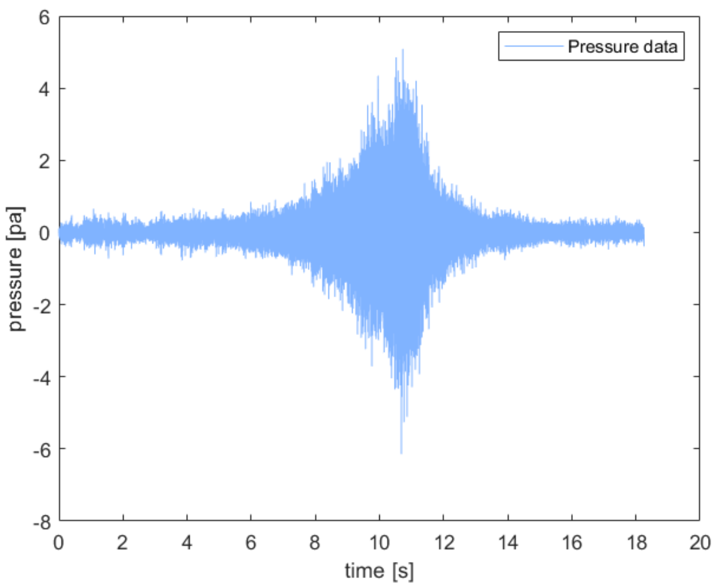


Figure 2.1: Aircraft Flyover - signal[Pa] vs time[s] plot

Within Figure 2.1 pressure measurements (in pascal) can be viewed over time. It can be observed that the highest pressure magnitudes can be observed around 10 to 11 seconds.

2.2. Question 2

Utilising the pressure measurements provided, a spectrogram could be constructed. Within this report the spectrogram function, from the signal processing toolbox, was utilised. As input the function first requires the data which should be represented, followed by the amount of data points within the chosen snapshot length. According to [3], a N in the range of 2048 to 8192 will provide sufficient result for the case of an aircraft flyover, sampled at similar frequencies. However, it is also mentioned to test a range of values to find a result which suits the analysis. The snapshot length is a subset of the full data set. Over each of these sub-elements/ snapshots a Fourier transform is performed to analyse the frequency content of the snapshot. When recombining all the snapshots again, the full signal can be analysed. This immediately show that a trade-off should be made between time and frequency resolution. Since talking small time steps (low N, high time resolution), will lead to limited values within the snapshot and therefore limited frequency resolution.

The third required input provides information regarding the overlap of the snapshots. Within the scope of this question and the current assignment, no overlap was utilised. The fourth value represent the size of the data points within the snap shots, summed with a possible padding. Zeros can be added to each of the snapshots in order to provide a smoother representation of the signal's frequency content. Here, the padding was set to 0 similarly to the overlap. Lastly, the sample-frequency is provided as final input to the function.

Below the spectrogram for the aircraft flyover can be retrieved, with the snapshot length(T) set to $0.075\,\text{s}$ and the overlap and the padding set to zero. From the utilised snapshot length the number of data points (**N**) is found to be 3000

$$T = N * \frac{1}{f_s} \Rightarrow N = T * f_s = 0.075\,\text{s} * 40\,\text{kHz} = 3000 \tag{2.1}$$

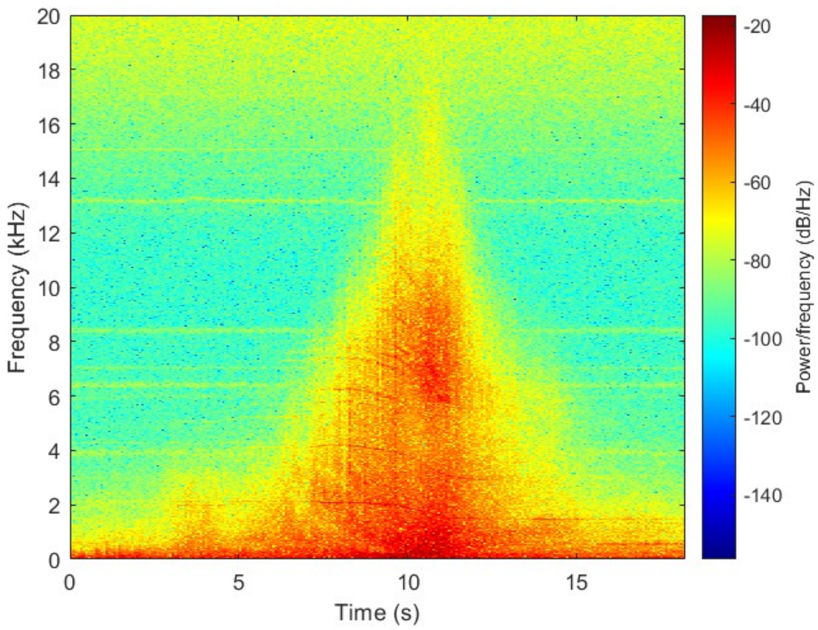


Figure 2.2: Spectrogram for an aircraft flyover with N=3000 (t=0.075)

section 2.8 includes two additional spectrograms, one with a large snapshot size t=0.25s and one with a small snapshot size t=0.01s. This clearly highlights the importance of the snapshot choice since a large snapshot, increase frequency resolution at the expense of time resolution. The opposite is observed when decreasing the snapshot size, now the time resolution increases at the expense of frequency resolution.

2.2.1. Observations

Within Figure 2.2 a range of observations can be made. Firstly, a Doppler shift can be observed. Around the 6 second mark the previously horizontal lines (representing the fan tones) curve downwards towards a lower frequency (due to the switch in relative velocity to the observer). Secondly, the key noise event can be observed at the 10 second mark. Here, the aircraft flyover is closest to the observer. Besides that it can also be observed that from 7 seconds to around 12 seconds, the power spectral density of the higher frequencies start to increase. Meaning that next to the low frequency noise, higher frequency noise now also is observed within the signal.

2.3. Question 3

To obtain the effective pressure Equation 2.2 was utilised. This expression was derived from [2] however a few modifications were made, to accommodate the current problem.

$$p_e^2 = \delta_f \sum_{r=0}^{N/2} 2P_r \tag{2.2}$$

When calling the Matlab function spectrogram, one of the resulting output data frames is the one sided power spectral density (thus only half the frequencies are considered (0-20000Hz) and the power is multiplied by two), as documented within the documentation of the signal processing toolbox[1]. As result the one-sided power spectral density (multiplication by two is done by the software) is summed up for half the sample frequency multiplied by the frequency resolution δ_f. After which the square root is taken, to obtain the effective pressure. This process is then repeated for each of 243 snapshots to obtain the effective pressure trough out the full aircraft flyover.

In Figure 2.3 the effective pressure can be seen superimposed on the Figure 2.1.

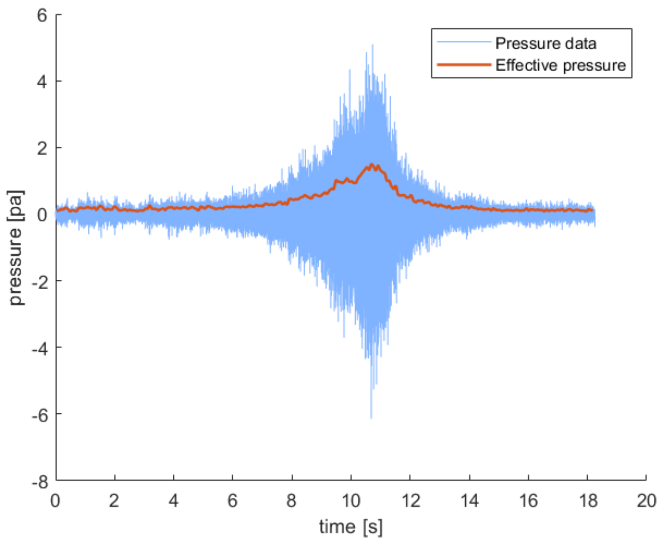


Figure 2.3: Effective pressure as a function on time, combined with the measured pressure by the microphone

It can be observed that the effective pressure is positive throughout the full duration of the signal. However it maintains significantly below the maximum detected pressure. Which is expected when comparing to the standard relation between the maximum pressure and the effective pressure ($p_{max}/\sqrt{2} = p_e$).

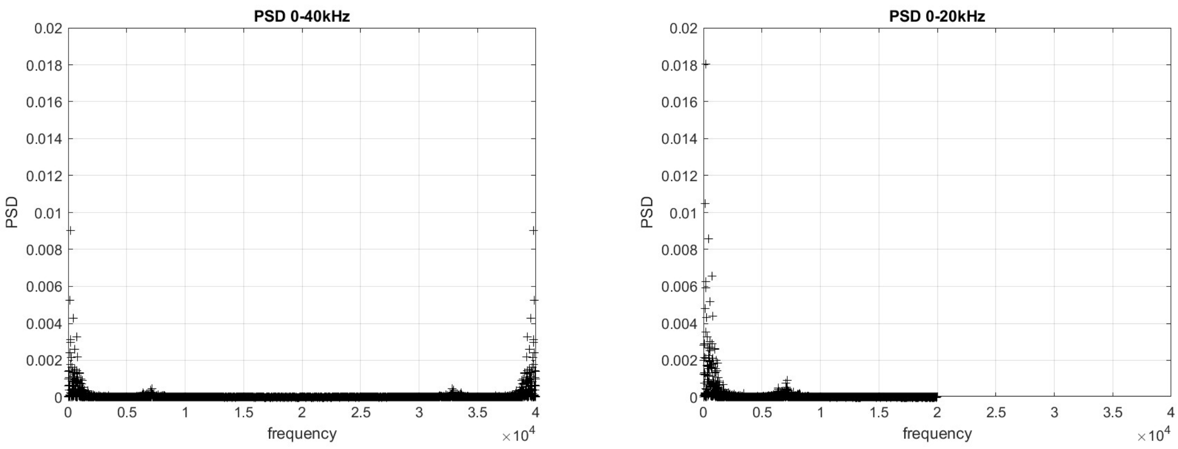
2.4. Question 4

To begin with, our snapshot size is now 0.125 s. Next, we create a time array in order to identify the index for which the signal is between 10.500 s and 10.625 s. This occurs at index, id = 85 (out of 146). Next, from the initial flyover data, y, all the pressure data points that were measured during this time are selected and Fourier transformed.

The power spectral density, PSD, is measured using the equation from slide 48[2]. Here X are the Fourier coefficient (note the amount of Fourier coefficients, is equal to N). Finally Delta(Δ) is equal to $1/f_s$.

$$P_r = \frac{|X_r|^2 (\Delta)^2}{T} \tag{2.3}$$

The results are plotted for the complete frequency range, 0 - 40 kHz. A second plot is created for half the frequency range, 0 - 20kHz. This is done by multiplying the initial power spectral densities by a factor of $(\frac{1}{\sqrt{2}})^2$ $*2^2$.



(a) range 0 - 40[kHz]

(b) range 0 - 20[kHz]

Figure 2.4: Power Spectral Density vs Frequency

2.5. Question 5

Previously within section 2.4 the one sided and two sided power spectral density were calculated. Within this section the procedure utilised to find the one sided power spectral density will be utilised to find the instantaneous OSPL expressed in dB. Furthermore an alternative procedure will be utilised which utilises time domain signals instead of the data expressed in frequency. The procedure will be described for one snapshot however within the Matlab code chapter 3, this procedure is repeated 146 times, to obtain a figure for the full aircraft flyover.

First a snapshot is made of length 0.125 s (N = 5000), then a Fourier transform is performed of this snapshot. Now utilising the power spectral density found previously the following equation can be utilised. Only now however the PSD is multiplied by a factor 2 and only the frequencies up to $20\,000$ Hz are considered.

$$OSPL = 10\log_{10}\left(\sum_{r=0}^{N/2} \delta_f \frac{2P_r}{p_{e,0}^2}\right) \tag{2.4}$$

With $\delta_f = \frac{1}{T}$, where T is the snapshot length expressed in seconds. $p_{e,0}$, the reference effective pressure is $2 * 10^{-5}$Pa.

Similarly to the procedure in the frequency domain, in the time domain the first step is to retrieve a snapshot of length 0.125 s (N = 5000) of the pressure signal (y). However now the following equation can be immediately utilised to obtain the OSPL using the snapshot (y_T).

$$OSPL = 10\log_{10}\left(\frac{\sum\frac{y_T\nabla}{T}}{p_{e,0}^2}\right) = 10\log_{10}\left(\frac{\sum\frac{y_T}{Tf_s}}{p_{e,0}^2}\right) = 10\log_{10}\left(\frac{\sum\frac{y_T}{N}}{p_{e,0}^2}\right) \tag{2.5}$$

Below it can be observed that irrespective of the method utilised, identical results are obtained for the OSPL over time. It can be seen that over time the aircraft flies overhead, resulting in an increase in OSPL, followed by a decrease in OSPL.

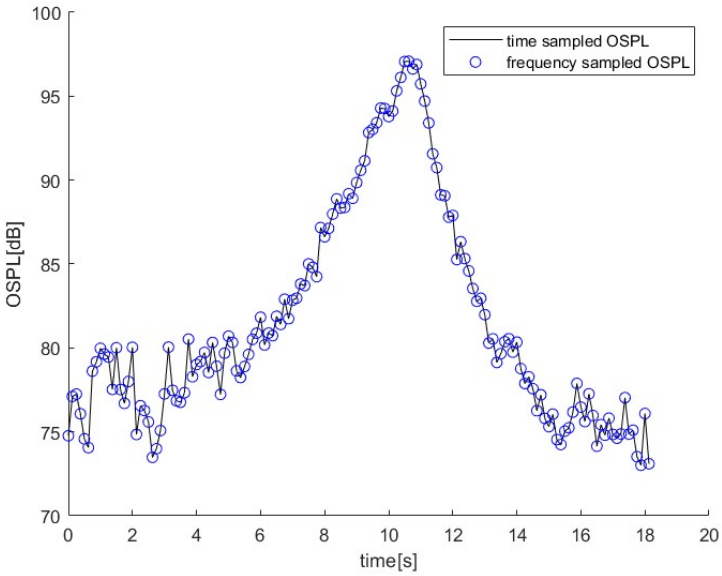


Figure 2.5: Instantaneous OSPL obtained trough sampling within the frequency and the time domain

2.6. Question 6

The human ear will perceive signals less loud depending on the frequency of the signal as explained in [3]. Therefore weighting factors can be utilised to compensate for this frequency sensitivity. The frequency sensitivity is modelled by Equation 2.6.

$$\Delta L_A = -145.528 + 98.262 * \log 10(f) - 19.509 * (\log 10(f))^2 + 0.975 * (\log 10(f))^3 \tag{2.6}$$

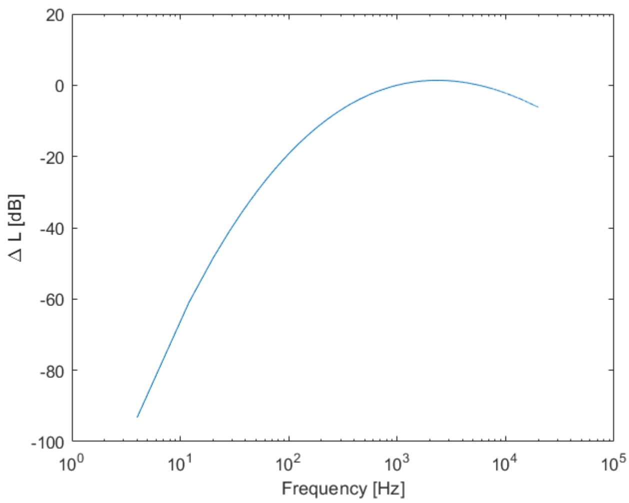


Figure 2.6: A weighting factor in dB as a function of frequency

This A-waiting Figure 2.6 is then applied to the on-sided power spectral density obtained previously ($2P_r$). Again similarly like the previous to the previous procedures, this is done independently for the frequency content of each snapshot. Then by applying the following two relations the A-weighted instantaneous sound pressure level can be computed, as observed in Figure 2.7.

$$L_A(r) = 10\log_{10}\left(\frac{2P_r}{p_{e,0}^2}\right) + \Delta L_A(r) \tag{2.7}$$

$$\text{OASPL} = 10\log_{10}\left(\delta_f \sum_{r=0}^{N/2} 10^{\frac{L_A(r)}{10}}\right) \tag{2.8}$$

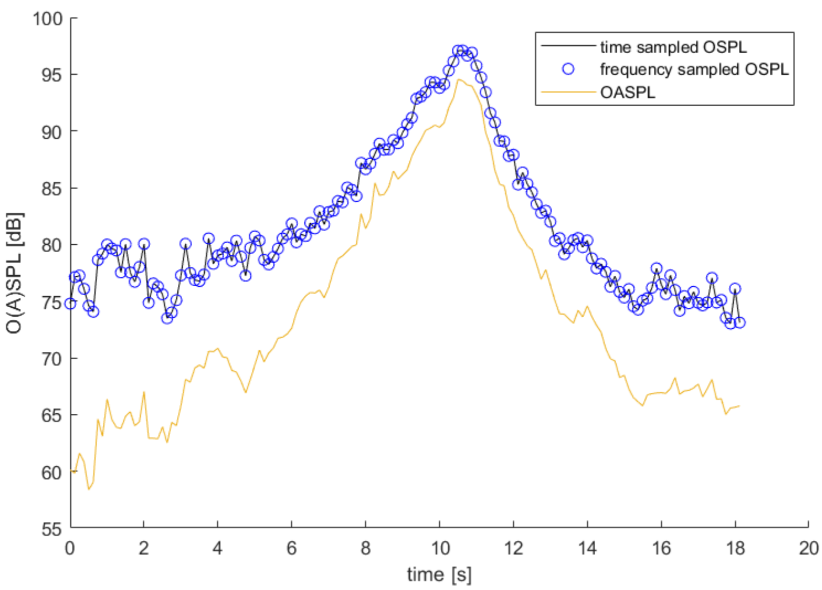


Figure 2.7: Instantaneous OSPL compared to OASPL

Due to the A weighting it can be observed that the instantaneous sound pressure level decreases for the full flyover. However a significant decrease is found at the beginning and the end, due to the large fraction of low frequency noise during these time periods (as could be observed in Figure 2.2). These lower frequencies are weighted more heavily since the human ear is less sensitive to low frequency noise, due to which these types of signals are perceived less loud.

2.7. Question 7

The Sound exposure level (SEL), expressed in dBA, is found by using Equation 2.9, where dt is the time snapshot and T_1 equals 1 (by definition). The integration was performed over all A-weighted SPL values which are larger than $L_A(max)$-10dbA.

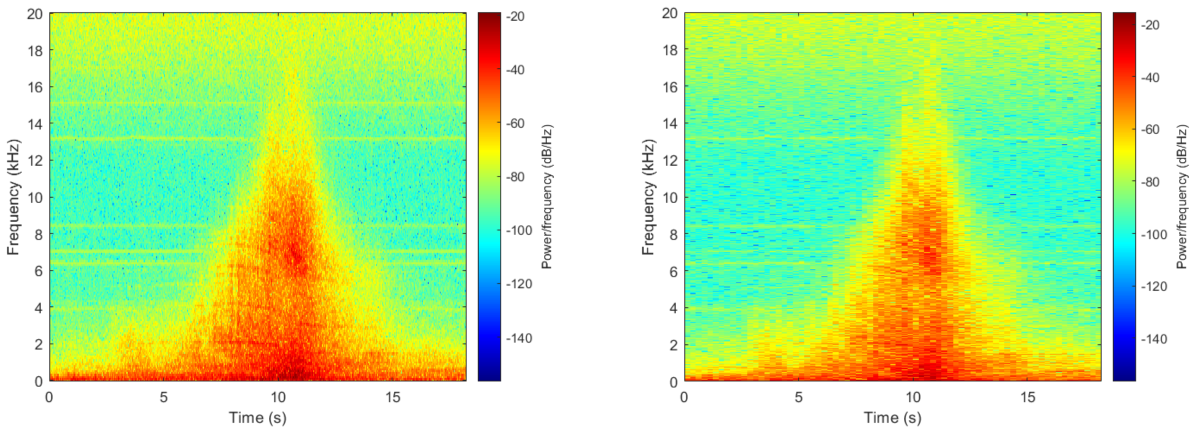
$$SEL = 10 \log \left[\frac{1}{T_1} \int_0^T 10^{\frac{L_A(t)}{10}} dt \right] \tag{2.9}$$

To allow for implementation within Matlab, Equation 2.9 was first discredited. Instead of utilising the integral, a summation was performed, to approximate the integral. This was done for the same bounds as explained previously.

$$SEL = 10 \log \left[\frac{1}{T_1} \sum_0^T 10^{\frac{L_A(t)}{10}} dt \right] \tag{2.10}$$

Implementing the above formula in Matlab results in an SEL of 95.843[dBA]. This is logical as it is higher than the $L_A(max)$ (= 94.556[dBA]). Furthermore it was noticed that when changing the bounds of the integral (implemented as summation within Matlab), the SEL increases to 96.322[dBA].

2.8. Additional Spectrograms



(a) Spectrogram for an aircraft flyover with N=400 (t=0.01)

(b) Spectrogram for an aircraft flyover with N=10000 (t=0.25)

Figure 2.8: Spectrogram for an aircraft flyover with varying snapshot size

References

[1] The MathWorks Inc. *spectrogram*. Natick, Massachusetts, United States, 2023. URL: `https://nl.mathworks.com/help/signal/ref/spectrogram.html`.

[2] Prof. dr. ir. M. Snellen. *AE4463 background assignment1 Acoustic Signal Analysis*. 2023.

[3] Prof. dr. ir. M. Snellen Prof. dr. D.G. Simons. *An introduction to general acoustics and aircraft noise*. 1th ed. Delft, Netherlands: TU Delft, 2023.

[4] Prof. dr. ir. M. Snellen Prof. dr. D.G. Simons. *Digital Signal Processing an introduction*. Delft, Netherlands: TU Delft, Dec. 2023.

3
Matlab code

```
%% Assignment 1: Advanced Aircraft Noise
% By: Elisabeth and Joshua

clear;

load('aircraft_flyover_microphone_assignment1.mat');
y = aircraft_flyover_microphone_assignment1;

t_begin = 0;
samplefrequency = 40000;
time_resolution = 1 / samplefrequency;

t_end = time_resolution*length(y) - time_resolution;

t = t_begin:time_resolution:t_end;

%% Part I: Figure 1 --> pressure over time
figure();
plot(t,y, ...
    Color="#80B3FF")
xlabel('time␣[s]')
ylabel('pressure␣[pa]')
legend("Pressure data")

%% Part II

% time resolution
time_reso = 0.075;   % 0.05 seconds time resolution
padding = 0;

N = time_reso*samplefrequency;
freq_resolution = 1 / time_reso;

figure(2);
% Second value represent the steps whcha re taken to analyse the data,
% larger values will lead to bigger blocks in the time axis, but smaller in
% the frequency axis (note: equally true when making thee value smaller)

% the fourth value represent the amount of zeros wich are added to the
% steps containing the seconds input amount of data. when the value is the
% same no padding will be added
spectrogram(y, N, 0, N+padding, samplefrequency, 'yaxis')
colormap jet

%% Part III
% Extract values from the plot, generated previously in part II
[S, F, T, P] = spectrogram(y, N, 0, N+padding, samplefrequency, 'yaxis');

pe = sqrt(freq_resolution.*sum(P));


figure();
hold on
plot(t,y, ...
    Color="#80B3FF")
plot(T, pe, ...
```

```
    Color="#D95319",...
    LineWidth=1.5)
xlabel('time␣[s]')
ylabel('pressure␣[pa]')
legend("Pressure data", "Effective pressure")



%% Part IV
time_reso = 0.125;   % 0.125 seconds
padding = 0;

N = time_reso*samplefrequency;
freq_resolution = 1 / time_reso;
t_array = 0:time_reso:t_end;
% index 85 returns 10.500 seconds to 10.625 seconds
id = 85;

pressure_85 = y(1 + (id-1)*N: (id)*N);
fourier_coef = fft(pressure_85);
Y = fourier_coef;

f = 0: freq_resolution: (N-1)*freq_resolution; %create the frequency x-axis
psd = (time_resolution^2/time_reso(end))*(abs(Y).^2);

figure(5);
plot(f, psd, '+k'); xlabel('frequency');
ylabel('PSD'); title('PSD␣0-40kHz'); grid; axis([0 samplefrequency 0 0.02])

% Half sided

f_half = 0:freq_resolution:(N-1)*freq_resolution/2;
psd_half = ((1/sqrt(2))^2) * (2^2) * psd(1:length(f_half));

figure(6);
plot(f_half, psd_half, '+k'); xlabel('frequency');
ylabel('PSD'); title('PSD␣0-20kHz'); grid; axis([0 samplefrequency 0 0.02])

%% Part V

OSPL_time = zeros(1,146);
OSPL_freq = zeros(1,146);

peo2 = (2*10^(-5))^2;

for id=1:1:146
    pressure = y(1 + (id-1)*N: (id)*N);
    pressure_fft = fft(pressure);

    psd = (time_resolution^2/time_reso)*(abs(pressure_fft).^2);

    pe2_time = sum(pressure.^2)*time_resolution / time_reso;
    OSPL_time(id) = 10*log10(pe2_time/peo2);

    pe2_freq = freq_resolution*sum(2*psd(1:2500));
    OSPL_freq(id) = 10*log10(pe2_freq/peo2);
end

% pe2_time = sum(pressure_85.^2)*time_resolution / time_reso;
% OSPL_time = 10*log10(pe2_time/peo2);
%
% pe2_freq = freq_resolution*sum(2*psd(1:2500));
% OSPL_freq = 10*log10(pe2_freq/peo2);

figure(7);
hold on
plot(t_array, OSPL_time, "-k")
plot(t_array, OSPL_freq, "ob")
legend("time sampled OSPL", "frequency sampled OSPL")

%% Part VI
```

```
% Procedure: To be used in for Part VI
% f(1:2500) <-- only half of the frequencies are considered
% for all of these frequencies the dla can be found
% dLa = -145.528 + 98.262 * log(f) - 19.509 * (log(f))^2 + 0.975 * (log(f))^3;
% La = PSL - dla
% where PSL = 10 log (2Pr/peo2)
% then oaspl = 10 log (deltaf * sum (10^La/10) ) Only performed over half
% of the frequencies due to doubling of Pr this takes into acount the equal
% peak at the negative side.

OASPL = zeros(1,146);

peo2 = (2*10^(-5))^2;

f_range = f(1:2500) + 4;

dLa = -145.528 + 98.262 * log10(f_range) - 19.509 * (log10(f_range)).^2 + 0.975 * (log10(f_range)).^3;

figure(9);
semilogx(f_range, dLa)
xlabel('Frequency [Hz]')
ylabel('\Delta L [dB]')

for id=1:1:146
    pressure = y(1 + (id-1)*N: (id)*N);
    pressure_fft = fft(pressure);

    psd = (time_resolution^2/time_reso)*(abs(pressure_fft).^2);

    pslr = 10*log10(2*psd(1:2500)/peo2);

    la = pslr + dLa;

    OASPL(id) = 10*log10(freq_resolution*sum(10.^(la/10)));
end

figure(8);
hold on
plot(t_array, OSPL_time, "-k")
plot(t_array, OSPL_freq, "ob")
plot(t_array, OASPL)
xlabel('time [s]')
ylabel('O(A)SPL [dB]')
legend("time sampled OSPL", "frequency sampled OSPL", "OASPL")

%% Part VII

% Perform integration with over bounds with 10db down time
% wher T1 is equal to one

LaMax = max(OASPL);

db_down_time = 10;

OASPL_10 = OASPL(OASPL > LaMax - db_down_time);
% Selected_DBa_time = t_array(OASPL > LaMax - db_down_time);
%
% figure();
% hold on
% plot(t_array, OASPL)
% plot(Selected_DBa_time, Selected_DBa)

SEL = 10*log10(sum(10.^(OASPL_10/10))*time_reso);
```