

# The sound of turbulence

by

LILY XU

**THESIS**

*for the degree of*

**MASTER OF SCIENCE**

(MASTER I ANVENDT MATEMATIKK OG MEKANIKK,  
RETNING COMPUTATIONAL SCIENCE)



DEPARTMENT OF MATHEMATICS  
FACULTY OF MATHEMATICS AND NATURAL  
SCIENCES  
UNIVERSITY OF OSLO

MAY 2015

MATEMATISK INSTITUTT  
DET MATEMATISK- NATURVITENSKAPELIGE FAKULTET  
UNIVERSITETET I OSLO



# Abstract

In fluid mechanics, much research has gone into finding out more about aneurysms using numerical simulations, but there are not many experiments that support these simulations. This has been due to the fact that models of aneurysms are hard to make, and that turbulence is difficult to detect when there are no access to intrusive techniques. In turbulent motion, sound signals with different frequencies are generated from the fluctuating velocities. Because of this, the fluctuating velocities may be detected using microphones. In this master thesis, the possibility to measure the sound of turbulence and detecting real physics is investigated. Known turbulent flows were recorded using a microphone, and their power spectra were compared with the turbulent spectra that had been found for these turbulent flows using intrusive methods such as particle image velocimetry, particle tracking velocimetry or hotwire. Measuring turbulence with sound is sensitive to noise since the sound level of the turbulent noise is not very high compared to the every day background noise. In addition to measuring the sound of turbulence, an extensive examination of the measuring equipment was done to validate the experimental setup.



# Acknowledgements

First of all, I would like to thank Atle Jensen and Andreas Austeng, for agreeing to work together and create this assignment for me, and for all the help and guidance both with the experimental work and the writing. This all started after me randomly taking a course in experimental fluid mechanics and discovering that there were a lot of signal processing involved in the experiments in the lab. However, other than that course, vector calculus was the closest to fluid mechanics I'd been, so it's safe to say that I did not have that much experience before I started this thesis. So I would probably have been completely lost without the weekly meetings and the recommended reading material.

A special thanks to the lab engineer at the Mathematics Department, Olav Gundersen, for helping me with the measurements, and for being there to help when LabView didn't want to do as I wanted, which it turned out was quite often.

A thanks must be addressed to my parents. To my father, whom without I would not have had become interested in fluid mechanics and taken the experimental fluid mechanics course in the first place. And to my mother, for the endless moral support and for reminding me every week that I could always come home if I needed a change of scenery.

Thank you Biørneblæs and the University Symphony Orchestra, for keeping my sanity and reminding me to take breaks and to also have some fun. Thank you Everlasting Sunshine for all the support.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	First set of experiments . . . . .	2
<b>2</b>	<b>Background</b>	<b>5</b>
2.1	Turbulent flow . . . . .	6
2.1.1	Power spectrum of a turbulent flow . . . . .	6
2.2	Signal theory . . . . .	7
2.2.1	Analogue and digital signals . . . . .	7
2.2.2	Generating sine signal with constant number of periods	8
2.2.3	Fast Fourier Transform to amplitude for sine signals . .	8
2.2.4	Estimation of the power spectrum . . . . .	9
<b>3</b>	<b>The experimental setup</b>	<b>11</b>
3.1	Amplifier . . . . .	14
3.2	Microphone . . . . .	15

<b>4</b>	<b>Validation of the components</b>	<b>17</b>
4.1	Amplifier . . . . .	17
4.2	Microphone . . . . .	22
<b>5</b>	<b>Sound measurements in turbulent pipe flow</b>	<b>27</b>
5.1	Post processing in MATLAB . . . . .	28
<b>6</b>	<b>Results and discussion</b>	<b>29</b>
6.1	Qualification of the amplifier and microphone . . . . .	29
6.1.1	The amplifier . . . . .	29
6.1.2	The microphone . . . . .	34
6.2	Results from the sound measurements . . . . .	39
<b>7</b>	<b>Conclusion and further work</b>	<b>45</b>
<b>A</b>	<b>Settings for the measurement &amp; Automation Explorer in LabView</b>	<b>47</b>
A.1	Validation of the different components . . . . .	47
A.1.1	Amplifier . . . . .	47
A.1.2	Microphone . . . . .	47
A.2	Sound measurements in turbulent pipe flow . . . . .	50
<b>B</b>	<b>LabView program for sound measurements</b>	<b>53</b>
<b>C</b>	<b>MATLAB files</b>	<b>55</b>



*CONTENTS*

III

C.1	Qualification of the amplifier . . . . .	55
C.1.1	amptest.m . . . . .	55
C.1.2	datafull.m . . . . .	63
C.1.3	amp.m . . . . .	64
C.2	Qualification of the microphone . . . . .	66
C.2.1	mictest.m . . . . .	66
C.2.2	sin1030.m . . . . .	68
C.3	Sound measurements . . . . .	72
C.3.1	welchpsd.m . . . . .	72
C.3.2	normpsd.m . . . . .	72
C.3.3	nyedata.m . . . . .	75
C.3.4	repetivity.m . . . . .	77



# List of Tables

3.1	The wire connections for the amplifier test in more detail. mic $\pm$ is the amplifier input. amp+ is the amplifier output. . .	14
4.1	The 5 amplifications that were tested. . . . .	21
5.1	The pump frequencies that were measured. . . . .	27



# List of Figures

1.1	The power spectrum for the turbulent flow with pump frequency 20 Hz from the first set of experiments. . . . .	3
3.1	The amplifier and microphone used in the experiments. . . . .	12
3.2	The experimental setup for the pipe experiment. . . . .	13
3.3	The wires connected to the NI myDAQ for the microphone test. . . . .	15
4.1	An illustration of the signal path for the amplifier test. . . . .	18
4.2	Block diagram of the LabView program for the amplifier test for frequencies 150-10000 Hz. . . . .	19
4.3	Block diagrams for the LabView programs used in the test for the frequencies 1-300 Hz. . . . .	20
4.4	Illustration of the signal path for the microphone test. . . . .	23
4.5	Block diagram and front panel for the LabView program used to test the microphone. . . . .	24
6.1	The first 48000 data points for the data measured for the amplified effect of 10 times. . . . .	30
6.2	Comparison of the first 60 data points of the original sine signal obtained from the amplifier test. . . . .	30

6.3	The amplified effect for the frequencies 1-300 Hz (top), and the amplified effect for all the amplifications tested where 1-300 Hz and 150-10000 Hz are plotted together (bottom). . . .	31
6.4	The amplified effect for the lower 3 levels of amplification. . .	32
6.5	The amplified effect for all the amplifications tested. The plot is zoomed in on the frequencies where the two tests overlap. .	33
6.6	The amplified effect for the different amplifications for 1-300 Hz done in an earlier test of the amplifier. . . . .	34
6.7	The amplified effect for each frequency that was tested. Plotted in both with logarithmic scales (top) and non-logarithmic scales (bottom). . . . .	35
6.8	The recorded (raw) signal plotted together with the signal after filtering for the lowest frequencies. . . . .	36
6.9	Plot of the Fast Fourier Transform for the lowest frequencies before and after filtering to make sure that the wanted signal is not removed. . . . .	37
6.10	Comparison of the estimated power spectrum for the first and second set of measurements. . . . .	40
6.11	The power spectrum for the different data files for each pump frequency plotted together to check for reproducibility. . . . .	41
6.12	The estimated power spectrum for the second set of measurements, plotted together with the turbulent power profile. . . .	42
A.1	Input and output tasks for the amplifier test for 150-10000 Hz.	48
A.2	The tasks settings for the microphone test. . . . .	49
A.3	Details for the input and output tasks for the LabView program used to record the pipe flows. . . . .	51

B.1 Block diagram for the LabView program used to measure the  
pipe flows. . . . . 54





# Chapter 1

## Introduction

### 1.1 Motivation

There are not many who know about aneurysms, but for those who know about them, they are very scary. A rupture of an intercranial aneurysm can cause a stroke, or in worst case lead to sudden death. Much research has gone into finding out more about these aneurysms using numerical simulations, but there is still shortage when it comes to experimental support. The object for this thesis is to make a contribution to the experimental work.

Turbulence is difficult to detect when there is no access for intrusive techniques such as hotwire, particle image velocimetry or particle tracking velocimetry, and since none of these methods are ideal for measuring flow in the human body, a non-invasive method is desired. One possible non-invasive method is to measure the sound of the turbulent flow.

The turbulent flow in aneurysms is very complex, since the blood does not flow constantly, but in pulses. For this thesis, water was used to simulate blood flow, as it is easily available and that there is much knowledge about the behaviour of water flows in pipes. Hopefully it would be possible to add to the complexity of the experimental setup step-wise in the future, and to in the end have an experiment with pulsative flow through a model of an aneurysm and be able to measure real physics.

## 1.2 First set of experiments

To see if it was possible to detect turbulent flow with a microphone, a first set of experiments was conducted in the Hydrodynamic lab at the University of Oslo. A microphone was taped on to the 50 mm pipe in the lab, and several turbulent flows with different flow rates were sent through while recording with the microphone. The recordings from the microphone were amplified through an amplifier before being written to a file with LabView.

A plot of the power spectrum for one of the turbulent flows (with pump frequency 20 Hz) is shown in Figure 1.1. The turbulent flows that were sent through the tube were flows known from invasive methods of detecting turbulence in the Hydrodynamic lab. Compared to the expected results it is not possible to be completely sure whether turbulence was detected with the non-invasive method or not. Since it was not confirmed that turbulent flows were detected using this experimental setup, the purpose of this thesis changed. To get a better idea of why the results turned out as they did, some additional experiments needed to be performed to cover the basics and to be able to endorse the measuring equipment. This initial set of experiments will be presented and discussed more in detail later in this thesis.

The new main question raised through this thesis is how to validate the experimental setup. Primarily if the amplifier and microphone is linear, and whether the microphone's frequency range covers all frequencies in question.

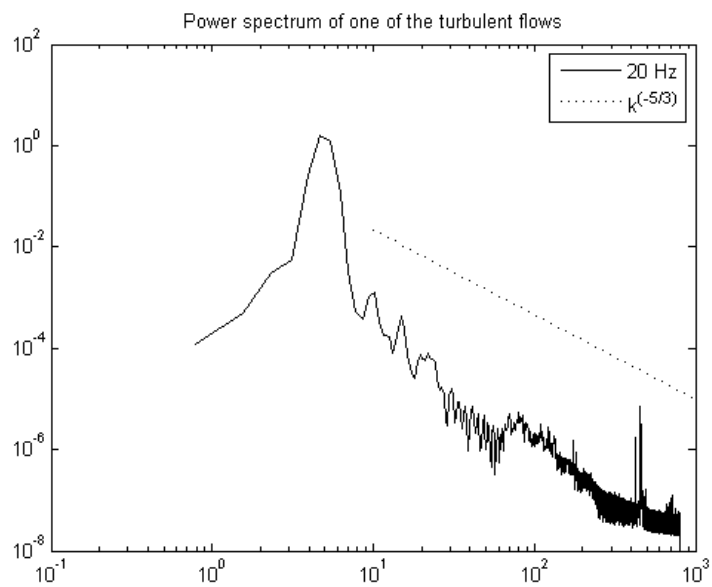


Figure 1.1: The power spectrum for the turbulent flow with pump frequency 20 Hz from the first set of experiments.



# Chapter 2

## Background

While measuring blood pressure, a sound can be heard between the systolic and the diastolic pressure. In short, these pressures marks the peak pressure and the minimum pressure in the arteries, respectively [1]. When listening through the blood pressure measuring device, the sound would get muffled somewhere between the two pressures. This muffle sound is believed to be from turbulent flow in the blood vessels because of the gradually opening of the vessels after being completely occluded. Studies have shown that the sounds that are heard while measuring blood pressure (Korotkoff sounds) is associated with turbulent flow [2].

If the Korotkoff sound could be associated with turbulent flow, the same hypothesis should be applicable to cases of partial obstruction of arteries where similar sounds (bruits) can be heard due to the disturbance of the blood flow, for example in regards to coronary artery diseases like atherosclerosis. By performing quantitative analysis of the sounds produced by blood flow, a study proposed a theoretical model that related the sound spectra measured from narrowed arteries with turbulent pipe flows [3], by assuming that the sound and turbulent spectrum were the same. A few years later, a study investigated the connection between these two spectra further. It was found that the sound spectra was different from the turbulence spectra for the same flow rates, but that the sound spectra could be used to diagnose artery diseases [4]. Some years later a study indicated that these muffle sounds, or murmurs, were in fact a consequence of turbulent flow [5].

Another disturbance of the blood flow are aneurysms. Through clinical stud-

ies, it has been confirmed that there should be turbulence in human intracranial saccular aneurysms [6].

One of the possible problems that could occur when trying to measure turbulent flow in blood is that the qualities of blood are more non-Newtonian when it comes to rheology [7]. The rate of dissipation for turbulent flow is only known for Newtonian fluids, so the known energy cascade may not be applicable for turbulent blood flow. However, a new method for determining whether a non-Newtonian flow is laminar, transitional or turbulent was found by relating non-Newtonian flows to Newtonian flows [8], so the knowledge of non-Newtonian flow is increasing.

In the rest of this chapter, some of the theory that is most relevant to this thesis will be presented.

## 2.1 Turbulent flow

Turbulent flow is a phenomenon that is presented as chaotic changes to the properties of flow. In turbulent flows there are unavoidably perturbations in initial and boundary conditions. Due to these perturbations, the turbulent flow fields display an acute sensitivity to such perturbations, and therefore no turbulent flows are identical. It is shown that turbulent flow also have perturbations depending on material properties [9]. However, there has been a significant progress in determining general turbulent flow characteristics such as the average velocity profiles and the degree of velocity fluctuations [10].

The velocity field of a turbulent flow can be seen as random, i.e. it does not have an unique value. This stems from the sensitivity to the perturbations mentioned above. Therefore, the turbulent flow might be described using statistic signal processing.

### 2.1.1 Power spectrum of a turbulent flow

Turbulent flows are made up by eddies, and through the eddies the kinetic energy is dissipated from the largest to the smallest scales. The power spectrum

(power spectral density) shows the second-order moment of a signal given in the frequency spectrum [11], and shows how the power is transferred between the frequencies [12]. Therefore, the dissipation should be seen in the power spectrum, and one way of characterizing a turbulent flow is through the energy cascade of this power spectrum. The energy cascade usually follows a specific energy density profile with a decay of  $-\frac{5}{3}$  [9, 13].

## 2.2 Signal theory

### 2.2.1 Analogue and digital signals

To be able to do digital signalprocessing on analogue signals, the analogue signals needs to be converted to digital signals.

One of the important things to notice when converting is the sampling rate. For the sampling rate, the sampling theorem should be followed to avoid misrepresentation of the analogue signals [14].

The sampling theorem says that in order to avoid any misrepresentation, the following inequality must be met:

$$F_s > 2 \cdot f_{max} \tag{2.1}$$

where  $F_s$  is the sampling rate and  $f_{max}$  is the highest signal frequency.

It is also important to note that if a signal with a given frequency is sampled with a sampling rate, but gets played with a different sampling rate, the played signal will not have the same frequency as the sampled signal. The signals frequency will then be either larger or lower depending on whether the sampling rate for playing is higher or lower than the sampling rate while sampling.

### 2.2.2 Generating sine signal with constant number of periods

The sampling frequency  $F_s$  of a signal is given by

$$F_s = \frac{n}{t} \quad (2.2)$$

where  $n$  is the number of samples, and  $t$  is the duration of the signal.

For each frequency,  $f$ , the number of periods,  $n_p$  is given by

$$n_p = t \cdot f \quad (2.3)$$

where  $t$  is the duration in seconds.

From equation 2.2, the number of samples,  $n$ , to generate a sine signal with a constant number of periods is given by

$$n = \frac{n_p}{f} \cdot F_s \quad (2.4)$$

### 2.2.3 Fast Fourier Transform to amplitude for sine signals

The Fast Fourier Transform (FFT) is an algorithm to compute the discrete Fourier transform [14].

The discrete Fourier transform (DFT) is in MATLAB implemented as [15]:

$$X_{DFT}(k) = \sum_{n=1}^N x(n) \omega_N^{(n-1)(k-1)} \quad (2.5)$$

where  $\omega_N = e^{(-2\pi i)/N}$ ,  $x(n)$  is the signal,  $i$  is the imaginary unit, and  $N$  is the number of samples.

The DFT has a corresponding inverse discrete Fourier transform

$$x(n) = \frac{1}{N} \sum_{k=1}^N X(k) \omega_N^{-(n-1)(k-1)} \quad (2.6)$$



The magnitude spectrum can be found by  $|X_{DFT}(k)|$ . For a sine signal with white noise, the Fourier transform can be seen as a Maximum Likelihood estimation for the amplitude [16]. The amplification can then be estimated by multiplying the value for the specific frequency with  $\frac{2}{N}$ , as the DFT is symmetric ( $|X_{DFT}(k)| = |X_{DFT}(N - k)|$ ) and there is no normalization in the DFT in the implementation that MATLAB uses.

### 2.2.4 Estimation of the power spectrum

The velocity field of a turbulent flow can be considered a random variable, and the turbulent flow itself could then be seen as a statistic random process [13]. If this is the case, it means that the measurements that were taken of the flows were finite sample sequences of a statistic random process. Since this finite sample is just a selection from the process, the power spectrum for the process can not be found from only the selection. However, it is possible to estimate the spectrum.

One of the ways to estimate the power spectrum is by using the Fourier transform of the measurements [16]. This method is good if there is only one frequency involved, but when there are more frequencies this method is prone to bias and variance. To minimize the variance, it is possible to split the measurement into overlapping sequences and Fourier transforming their estimated ensemble averages. This method is called the Welch's method, and is defined as [11]:

$$\hat{P}_W(e^{j\omega}) = \frac{1}{KLU} \sum_{i=0}^{K-1} \left| \sum_{n=0}^{L-1} w(n)x(n+iD)e^{-jn\omega} \right|^2 \quad (2.7)$$

where  $\hat{P}_W(e^{j\omega})$  is the estimate. Each sequence of length  $L$  is overlapping  $D$  points with the successive sequence. For  $N$  total data points,  $K$  is the number of sequences needed to cover all data points.  $U = \frac{1}{N} \sum_{n=0}^{N-1} |w(n)|^2$  and  $w(n)$  is a data window that can be applied to modify each sequence.  $j$  is the imaginary unit, and  $e^{j\omega}$  denotes the frequency in units of  $\pi$ .



# Chapter 3

## The experimental setup

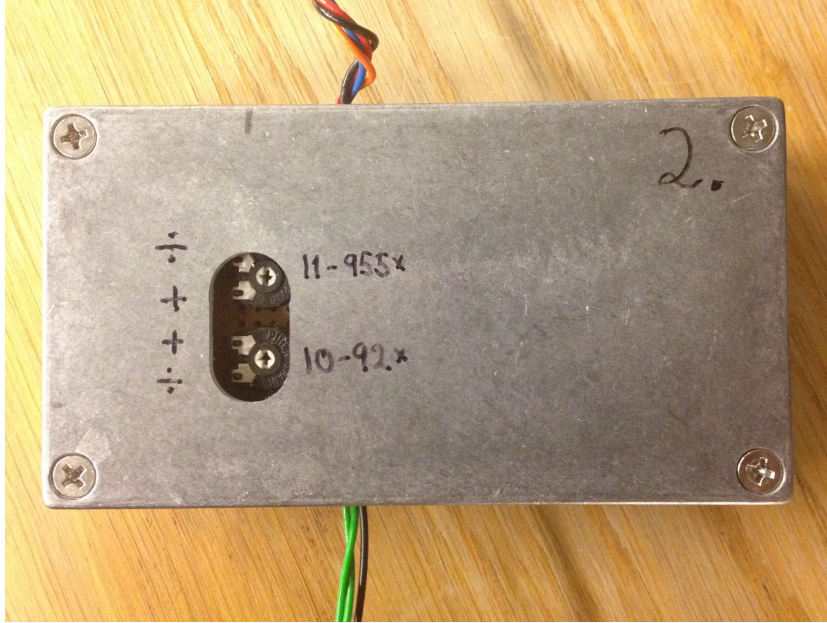
The measurements were taken in the Hydrodynamic lab at the Department of Mathematics at the University of Oslo.

To record the data a NI myDAQ from National Instruments, which is a data acquisition device used to measure and analyze signals [17], was used. Together with the device, National Instrument also has a system design software, LabView [18]. The NI myDAQ was connected via USB to a Sony Vaio laptop with LabView installed. From the NI myDAQ, a microphone was connected through an amplifier made in house.

Figure 3.1a and 3.1b shows the amplifier and microphone used for all the experiments in this thesis.

For the first set of experiments, the microphone was placed on the 50 mm flow loop in the Hydrodynamic lab. This setup is shown in figure 3.2.

As mentioned in the introduction, the results from the first set of experiments were inconclusive due to the uncertainties when it came to the effect that the measuring equipment had on the data sets. To be sure whether the results were caused by real physics or not, these had to be validated. New experiments were therefore performed to hopefully be able to validate the data obtained from the first set of experiments, and to be sure of what parts of the results might be influenced by the measuring equipment.



(a) The amplifier used in the experiments.



(b) The microphone that was used in the experiments.

Figure 3.1: The amplifier and microphone used in the experiments.

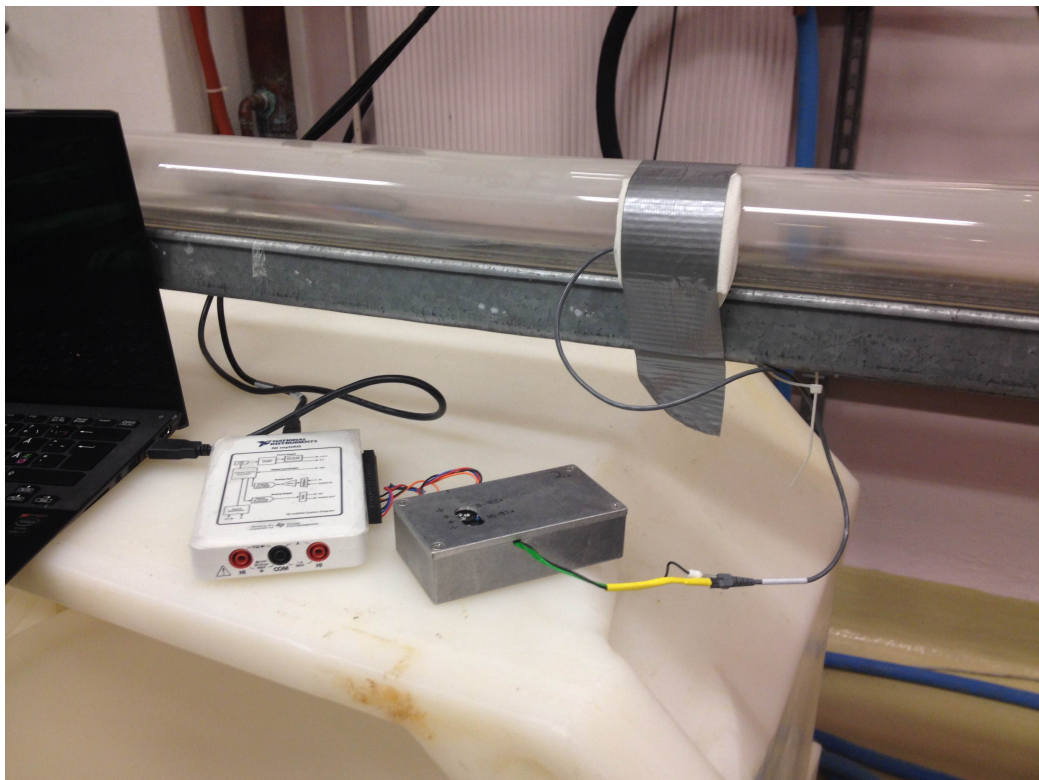


Figure 3.2: The experimental setup for the pipe experiment.

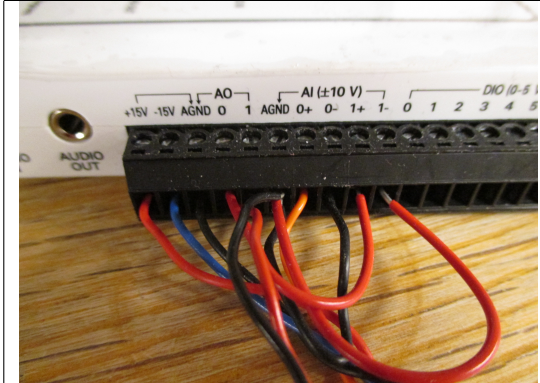
	<table> <tr> <td>AGND (AI)</td> <td>→</td> <td>AI 1-</td> </tr> <tr> <td></td> <td></td> <td>AI 0-</td> </tr> <tr> <td></td> <td></td> <td>mic-</td> </tr> <tr> <td>AO 0</td> <td>→</td> <td>mic+</td> </tr> <tr> <td></td> <td></td> <td>AI 1+</td> </tr> <tr> <td>AI 0+</td> <td>→</td> <td>amp+</td> </tr> </table>	AGND (AI)	→	AI 1-			AI 0-			mic-	AO 0	→	mic+			AI 1+	AI 0+	→	amp+
AGND (AI)	→	AI 1-																	
		AI 0-																	
		mic-																	
AO 0	→	mic+																	
		AI 1+																	
AI 0+	→	amp+																	

Table 3.1: The wire connections for the amplifier test in more detail.  $\text{mic}\pm$  is the amplifier input.  $\text{amp}+$  is the amplifier output.

The experimental setup was tested component-wise, by stripping down the setup and testing the components in steps. First, the microphone was disconnected, making it possible to test only the amplifier and also control that the signals were well represented with the NI myDAQ. Then the microphone was connected again, and the microphone was tested together with the amplifier. In other words, when testing the microphone, the whole experimental setup would be tested, and the results from the microphone test would say something about how the equipment effected the recorded signals.

### 3.1 Amplifier

To test the amplifier, the microphone was disconnected from the experimental setup. Instead, wires were connected from the amplifier's input to the NI myDAQ. As seen in figure 3.1a, the amplifier has two sets of wires connected to it, one on the top and one on the bottom. The top and bottom set represents the amplifier output and input, respectively. The amplifier was connected to the NI myDAQ in a loop, by connecting both sets of wires to the NI myDAQ. Table 3.1 shows the wire connections to the NI myDAQ in more detail, with an overview of the wires. In addition to the orange wire from AI 0+ to the amplifier output, the three wires to the left (+15V, -15V and AGND) are also connected to the amplifier output.

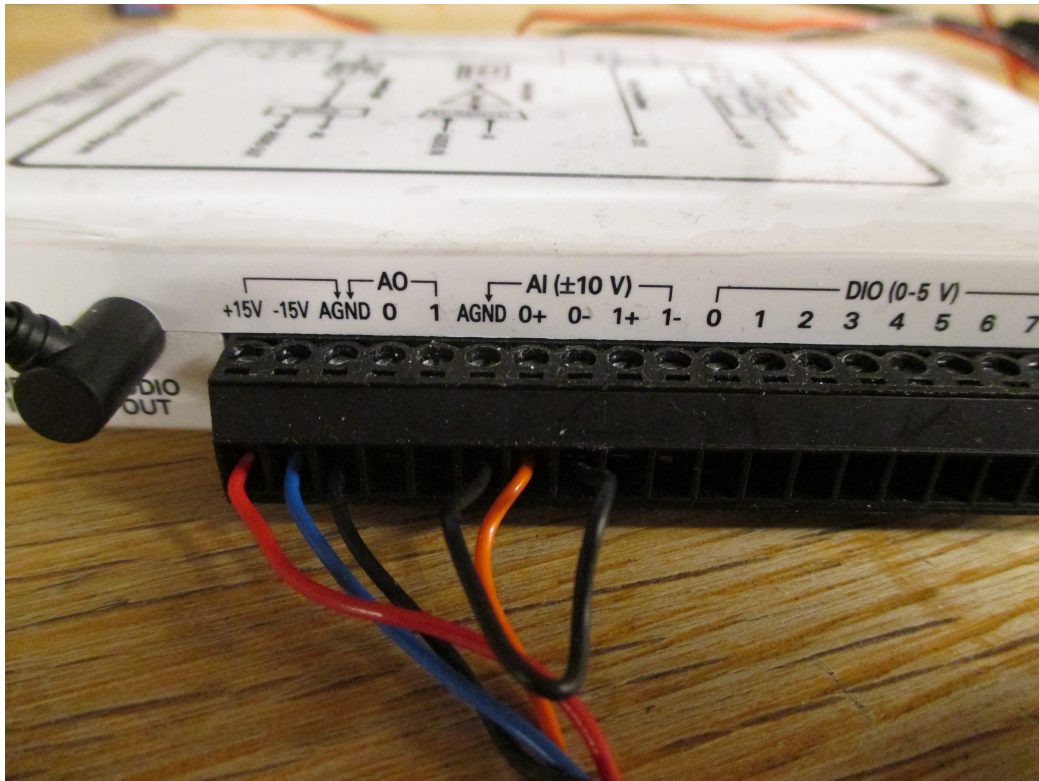


Figure 3.3: The wires connected to the NI myDAQ for the microphone test.

## 3.2 Microphone

From the amplifier test, the connected wires on the NI myDAQ needed to be switched. Figure 3.3 shows the updated wire connections. As seen in the figure most of the wires were removed from the amplifier test. Using the wire connections from table 3.1 as a starting point, the wires that went in a loop that were used to control the signal that went in the NI myDAQ were removed, and so were the two wires that were from the amplifier output. Instead a microphone was connected to the amplifier input, and a Boss headset was connected to "Audio out" to be used as the speaker. With the exception of the headset, this setup is technically the same as for the pipe experiment (the first set of experiments).

The microphone was taped on to a 1 mm plexiglass. The plexiglass was then placed between the two earpieces of the headset, and the earpieces were then fixed shut using plastic strips.





# Chapter 4

## Validation of the components

### 4.1 Amplifier

In the experimental setup, the analogue signal recorded from the microphone went through an amplifier before transferred into a digital signal. For the signal to be a true representation of the analogue signal, the amplifier needs to be linear, i.e. amplifying each frequency equally.

For illustrational purposes and to describe the amplifier, Figure 3.1a on page 12 shows the amplifier used in the experiments. The upper and lower switch controls the effect of the amplifier. During the test of the amplifier, it was discovered that the top and bottom text, "11-955x" and "10-92x", that shows the effect were not completely accurate. A more correct text for the amplifier would have been "1-9.55x" and "10-100x" for the top and bottom text, respectively, since having both the bottom and the top switch on maximum effect, the total amplified effect was 955x (or approximately 1000x). The bottom switch is the main switch for the amplifier, and the top switch is mainly used to add additional amplification. Therefore, the top switch on it's own does not have it's full effect when the lower switch is on it's minimum.

Figure 4.1 shows an illustration of the signal path for this experiment.

To check for linearity, a generated sine signal was sent out of the NI myDAQ, through the amplifier with frequencies ranging from 0 Hz to approximately

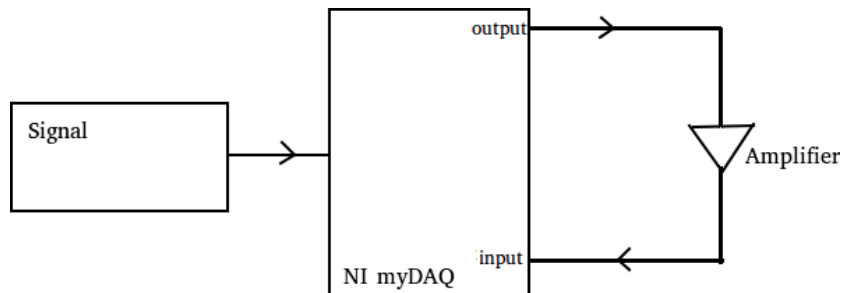


Figure 4.1: An illustration of the signal path for the amplifier test.

10000 Hz in 10 Hz step. The amplified signal was then sent back to the NI myDAQ. The original sine signal was also sent through the NI myDAQ to make sure that the sine was represented correctly and also to have better control of the signals. The amplified signal was compared to the original sine signal by plotting in LabView, and both the signals were then sent to a file for further analysis during the post processing.

From figure 1.1 on page 3 it would seem that the important frequencies were in the range of 1-300 Hz approximately, so an additional test for 1-300 Hz with 1 Hz step was also performed to get a better look at the lower frequencies. Both amplifier tests were programmed in LabView. For the frequencies 0-10000 Hz, the sine signals were generated in the same LabView program as the actual amplifier test. Figure 4.2 shows the LabView program used to test the frequencies 0-10000 Hz. Initially, this program also saved the frequencies from 0-140 Hz, but since it was desired to have 15 or more periods for each frequency to calculate the mean amplification, these frequencies were removed from the program because they would be unusable in the post processing.

For the settings in the Measurement & Automation Explorer in LabView for the LabView program, see appendix A.1.1.

The first amplifier test resulted in very large data files, seeing that every frequency was sampled with 10000 samples with a sampling frequency at 100000, which made the post processing of the data relatively slow. If the same program was used to test the frequencies 1-300 Hz, the lower frequencies would be unusable because they would not have enough periods. To both decrease the file sizes and to make sure that every frequency had 15 periods, another LabView program was written.

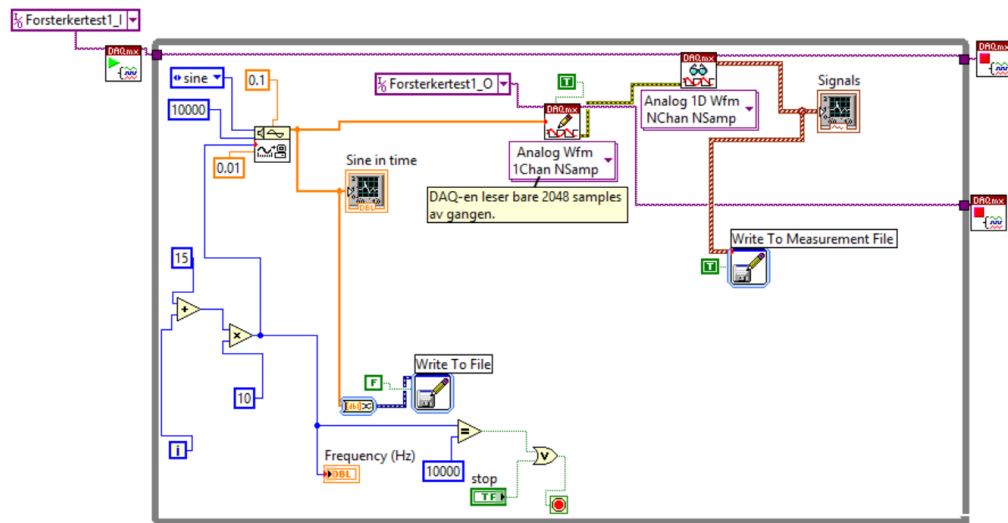
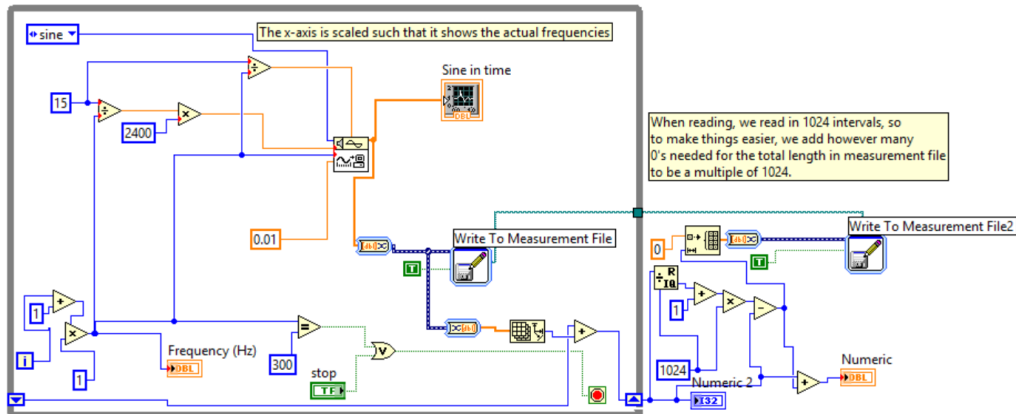


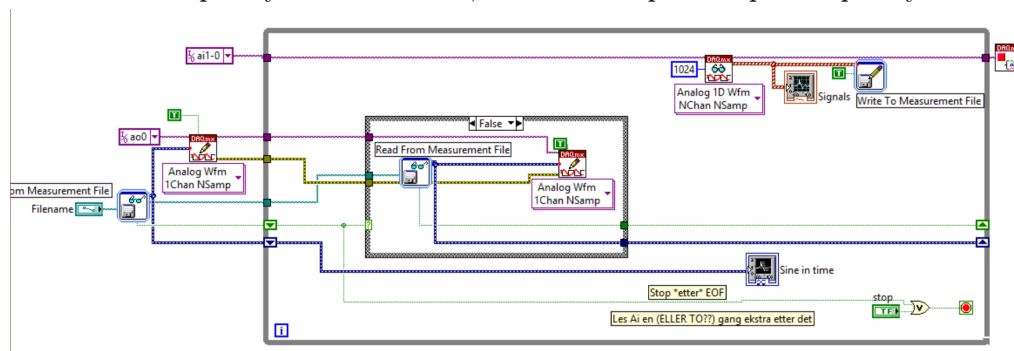
Figure 4.2: Block diagram of the LabView program for the amplifier test for frequencies 150-10000 Hz.

To make the post processing easier, a constant number of samples and periods per frequency would be ideal, but the NI myDAQ does not support a varying sampling frequency. Since the sampling frequency could not be changed while running the program, the constant number of samples and periods for each frequency would just be interpreted as the same signal by the NI myDAQ. For the NI myDAQ to be able to distinguish the different frequencies, the signal either needed to have varying number of periods or number of samples used to sample the actual frequency. This gave problems when the sine signal was generated in the same program as the amplifier test. To solve this problem, the generating of the sine signal was extracted and put in a new LabView program. The LabView program for generating the sine signal generated a file containing a chirp-like signal with frequencies 1-300 Hz with 1 Hz step. The amplifier test was then updated to read from this file. The program with the signal generator and the program with the updated amplifier test are shown in Figure 4.3a and 4.3b, respectively.

To make sure that every frequency has a constant number of periods, the parameters sent in to the built-in sine generator in LabView needed to be adjusted for every frequency. For this particular experiment, the desired number of periods per frequency was 15, and the sampling frequency  $F_s = 2400$  was used. Using equation 2.4 and 2.3 on page 8, the first frequency, 1 Hz, is sampled with 36000 samples, with a duration of 15 seconds, creating



(a) Block diagram of the LabView program for generating a data file with sine waves with frequency from 0-300 Hz, and with 15 periods per frequency.



(b) Block diagram of the LabView program for the amplifier test for frequencies 1-300 Hz.

Figure 4.3: Block diagrams for the LabView programs used in the test for the frequencies 1-300 Hz.

Amplified effect	Top switch	Bottom switch
10x	Min	Min
100x	Min	Max
1000x	Max	Max
11x	Max	Min
18x	Center	Center

Table 4.1: The 5 amplifications that were tested.

15 periods. The last frequency, 300 Hz is then sampled with  $\frac{36000}{300} = 120$  samples, with a duration of 0.05 seconds.

Because of how the NI myDAQ reads and handles the signals, the signal generator also had some additional code on the outside of the main loop, as seen in the figure. The purpose of this extra code was to add extra zeros to the file with the sampled sine waves to ensure that every frequency was recorded. Since the NI myDAQ had some delay when recording the sine signal and since it recorded a fixed number of samples at a time, it was a possibility that the last samples would not be recorded if the total number of samples were not a multiple of the sample rate. However, this additional code in the sine generator did not seem to have a significant effect on the result in comparizon with the sine signal file generated from the LabView program without this extra code.

Table 4.1 shows the 5 different amplifications that were tested in the amplifier test. As stated earlier in this subsection and also shown later in this thesis, the more accurate amplified effects would be 10x, 100x, 955x, 11x, 18x, respectively, but the following tables or references to the amplified effects will use the amplified effect shown in this figure.

### Post processing in MATLAB

For the obtained files from both tests, the mean amplification for a frequency was calculated by extracting the maximum and minimum points for each of the 15 periods. Since the minimum point would be negative, it was multiplied by  $-1$ , and then the mean was calculated with these values.

For the test for frequencies 0-10000 Hz, the duration of each sine signal that was generated had a duration of 0.1 seconds. As the number of samples was

constant, each frequency was represented by 10000 samples, and therefore each frequency would have a different amount of periods within the 10000 samples. From equation 2.3 on page 8, only the frequencies from 150 Hz could be used to calculate the amplification since the desired number of periods was set to 15. With the number of periods,  $n_p$ , each period for each frequency would then be represented with  $\frac{10000}{n_p}$  samples. For the frequencies larger than 150 Hz, only the first 15 periods were used.

For the second test (frequencies 1-300 Hz) the number of periods was constant at  $n_p = 15$  for each frequency, and the number of samples,  $n$ , per frequency was calculated with Equation 2.4 on page 8. The samples used to represent one period was then simply  $\frac{n}{15}$ .

In both tests there would be instances where the calculated sample number that was needed was not an integer. Because of this, a combination of rounding up, down or to the nearest integer was used to try to make sure that each period extracted included the real max and min point for the amplified sine.

The MATLAB files for calculating the mean amplification can be found in appendix C.1. Appendix C.1.1 is the main program, while appendix C.1.2 and C.1.3 are functions to find the mean amplification for the 0-10000 Hz and 1-300 Hz test, respectively.

## 4.2 Microphone

In addition to the amplifier, the microphone that was used had to be linear and be able to detect every frequency to be sure whether or not the results were physics or just an affect from the components in the setup. Testing the microphone was quite difficult compared to the amplifier, since there were more components that needed to be used. For the amplifier, the only unknown component was the amplifier, so the effects that were seen in the results had to be from the amplifier. For the microphone however, the sound to be recorded needed to be played such that the microphone had something to record, thus giving two unknown components. It may therefore not be possible to distinguish between the effect from the speaker and the effect from the microphone. Nevertheless, a test was run to have more control of the microphone that was used. The microphone was tested by playing sine signals with different frequencies using a speaker (Bose headset).

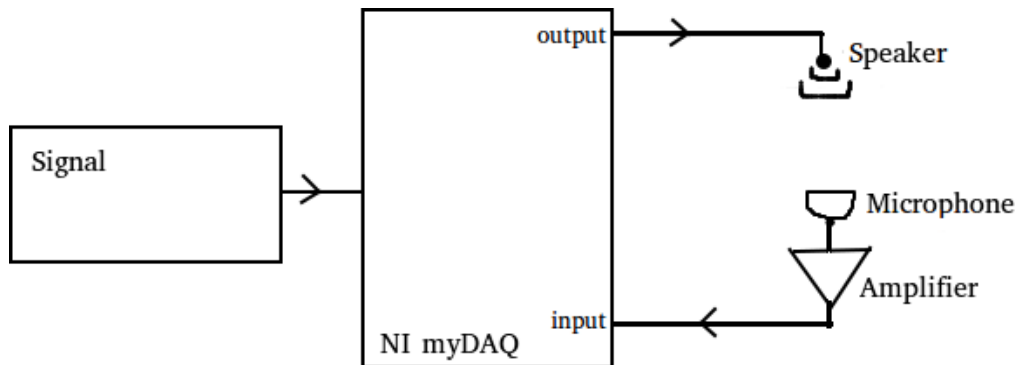


Figure 4.4: Illustration of the signal path for the microphone test.

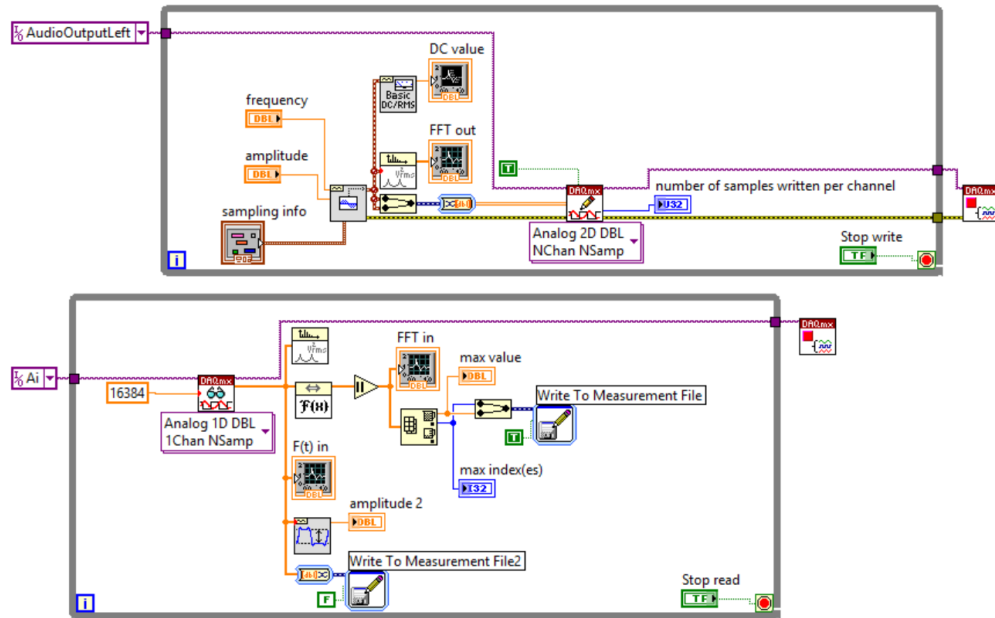
At first, the microphone was tested using the same programs as for the amplifier, since it in theory only was a change in the NI myDAQ's input and output settings. However, the speaker had problems playing the lowest frequencies, as expected. In addition, it seemed that the results from the tests were not as expected when the frequency was changed automatic with the LabView program. So another LabView program was made where the frequency could be altered by the user in the front panel while running the program. The LabView program used was written with the help of lab engineer Olav Gundersen from the Department of Mathematics.

Figure 4.4 shows an illustration of the signal path for the microphone test.

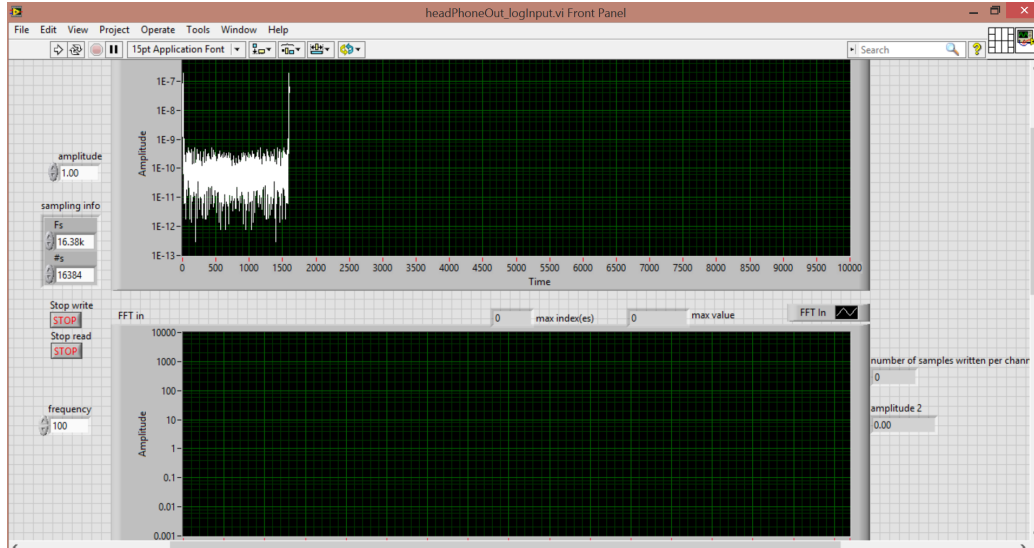
The type of headset used was a noise cancelling headset from Bose, and it was believed that the noise cancelling would reduce the background noise. However, it was a slight possibility that the system would affect the recorded sound, since the earpieces would send out cancelling sound waves. The earpieces were closed shut against each other with the microphone inside, so assuming that the noise cancelling system only sent waves from the outside of the earpieces the noise cancelling system should not have a significant affect of the recorded sound.

The settings in the Measurement & Automation Explorer in LabView that were used are described more in detail in appendix A.1.2.

Figure 4.5 shows both the front panel and the block diagram for the LabView program that was used. The program ran continuously, and for each loop the headset would play a sine signal with an amplitude and frequency given in the front panel of the program. The microphone would record the sound



(a) Block diagram of the LabView program used to test the microphone.



(b) Front panel of the LabView program used to test the microphone.

Figure 4.5: Block diagram and front panel for the LabView program used to test the microphone.



from the headset. The program would calculate the Fast Fourier Transform (FFT) of the recorded signal, and the maximum point in this FFT together with the index where the maximum point was located would be saved to a file. In addition, the signal recorded from the microphone was saved to another file, to allow the possibility to control the results obtained straight from the LabView program.

As the frequency could be changed while running the program, the settings for writing to file was changed such that the new data for each loop iteration would be appended to a given file. This was done to avoid having the same number of files as the number of frequencies that would be tested which would result in a quite complicated post processing. From the front panel it was easy to see the jump from one iteration to the other in the loop, so it was made sure that at least 15 iterations had been done before switching to the next frequency to ensure that there were enough points to calculate the mean amplification later in the post processing.

For the lower frequencies the background noise (mostly the "wall noise" at 50 Hz) would take over the actual signal, so the max point and its index given from the LabView program would not be the correct one. A quick test run was done beforehand to determine at how low frequency it was possible to record with no jumps in the max indices seen in the front panel. In this case the lowest frequency that didn't seem to get misinterpreted as 50 Hz was 30 Hz. Just in case there would be jumps, the frequencies were run from the highest to the lowest frequency such that if one of the lower frequencies would suddenly have a max point at a wrong index it would be easy to remove it while post processing. The program was then run with frequencies from 30-90 Hz with 10 Hz steps, from 100-400 Hz with 50 Hz steps, from 500-1000 Hz with 100 Hz steps, and 2000 Hz.

The program was run three more times with constant frequencies of 10, 20 and 30 Hz, to see if it was possible to filter out the higher frequencies from the recorded signal to find the amplification of the actual frequencies that we tested. By running the 30 Hz again it would be possible to compare the amplified value after filtering with the one that was found with the main method described above.

### Post processing in MATLAB

Finding the mean amplification for the frequencies larger than 30 Hz was quite simple, since the indices (the frequency) for each max point of the FFTs were saved. For each frequency, the mean max point was found, then the mean max point was converted to amplitude by the method explained in chapter 2.2.3 on page 8.

For the frequencies 10, 20 and 30 Hz, the main noise seemed to be the 50 Hz, so a simple filtering was done in the frequency domain by removing the high frequencies such that only the desired signal was left. This was a brutal way of doing it, as there in practice is not possible to make an ideal low pass filter which is what this method essentially was. However, while removing the frequencies it was made sure that there was a buffer of frequencies after the desired frequency that were not removed. The frequency in the transition was divided by 2 so that there would be a less abrupt transition between the kept frequencies and the ones that were completely filtered away. This was done to minimize the Gibbs effect on the filter, which occurs when reconstructing discontinuous signals [14]. After the signal was filtered, the filtered signal was transferred back to the time domain and controlled with the original signal. Then the amplification of the frequency was found in the same way as above, by using the max point of the FFT. However, only one max point was found from each data file rather than splitting the file up into parts and then calculating the mean from the different max points. This could have an effect on the results, but seeing that the max values did not seem to fluctuate much for any of the higher frequencies, it was assumed the values that were found still could give an idea of how the microphone or headset affected the signals.

The MATLAB codes for the post processing can be found in appendix C.2. Appendix C.2.1 is the main program, while appendix C.2.2 is the function to filter the lower frequencies 10, 20 and 30 Hz and to find their amplification.

# Chapter 5

## Sound measurements in turbulent pipe flow

The experimental setup was as explained in Chapter 3 and is shown in figure 3.2 on page 13. The microphone was placed on the 50 mm tube in the Hydrodynamic lab and 5 measurements were taken of flows with different pump frequencies. The LabView program used to measure the turbulent pipe flows was written by the lab engineer at the Department for one of his earlier experiments, and is shown in appendix B. In practice the program should be very similar to the one used to validate the microphone.

It was discovered that the length of the files were shorter than desired, and that it would be ideal to also be able to test for repetition. A new set of measurements were therefore taken, to both have more data points to work with and to make it possible to confirm that the measurements were not dependent on the time or date that they were taken. In this new set,

25th of June:	18th of August:
0 Hz	0 Hz
20 Hz	20 Hz
25 Hz	25 Hz
30 Hz	30 Hz
31 Hz	35 Hz

Table 5.1: The pump frequencies that were measured.

16 measurements were taken, where each pump frequency was measured at least 3 times. Table 5.1 shows the pump frequencies that were used in the two sets of measurements.

## 5.1 Post processing in MATLAB

For the first set of measurements, there was a technical error while measuring the flow with pump frequency 20 Hz. The data was appended to a file with data from the 25 Hz pump frequency. So some data points needed to be removed from the file in question so that the file only contained data from one pump frequency.

The MATLAB codes used for the post processing is found in appendix C.3.

To estimate the power spectral density the Welch's estimate was used. There was uncertainties whether or not the built-in function had some unwanted normalization built-in, so an own implementation of the Welch's estimate for the power spectral density was made and is found in appendix C.3.1.

The MATLAB code for finding the power spectrum for the first and the second set of measurements are found in appendix C.3.2 and C.3.3, respectively.

The program used to check for reproducibility can be found in appendix C.3.4.

# Chapter 6

## Results and discussion

### 6.1 Qualification of the amplifier and microphone

#### 6.1.1 The amplifier

The files obtained from both LabView programs for the amplifier test had three columns. The columns contained the sample number, the amplified signal and the original signal, respectively. The original signal was recorded to control that the signal that was sent through the LabView had the right amplitude, and since it was sent through the NI myDAQ and back, the signal had the same delay as the amplified signal.

Due to some buffer problems with the NI myDAQ, the sine signal obtained from the test of the amplifier for 1-300 Hz contained an extra top at the start. The top for the 10 times amplification is shown in figure 6.1, where the first 48000 data points are shown. As seen in the figure, the first frequency has 15 periods in addition to an extra top at the beginning. For both the original sine signal and the amplified sine signal for each amplification that was tested, the first top was removed before further processing. To make sure that the correct number of samples was removed from each set of sine signals, the original sine signals were plotted together. Figure 6.2 shows the first 60 data points of the original sine signals for each amplification.

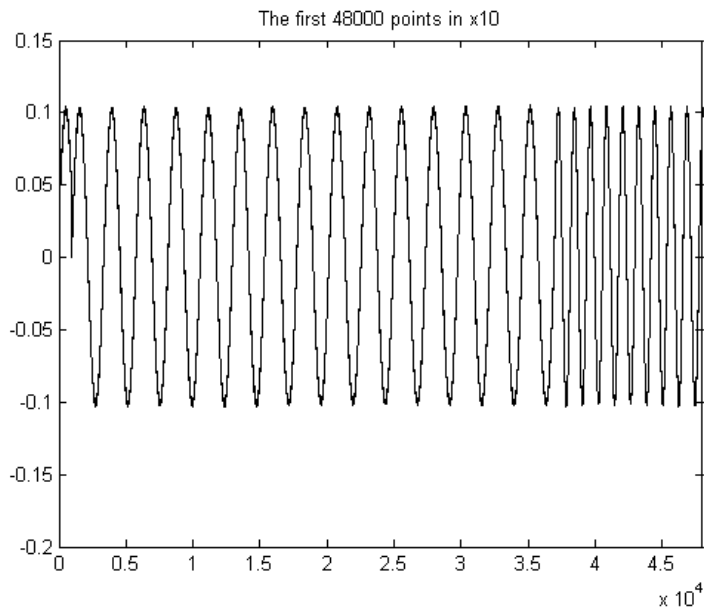


Figure 6.1: The first 48000 data points for the data measured for the amplified effect of 10 times.

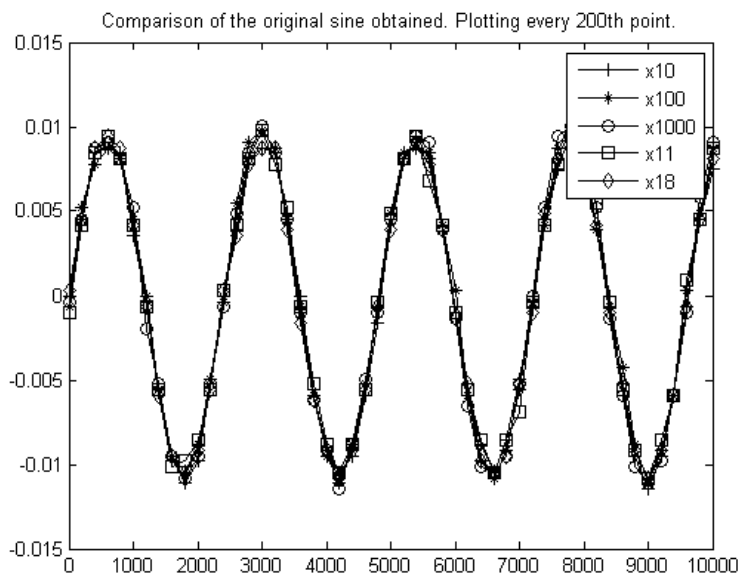
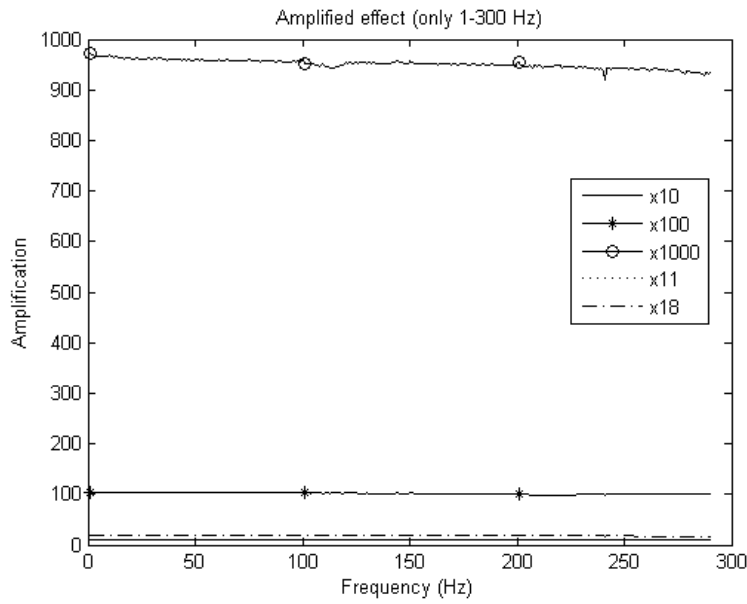
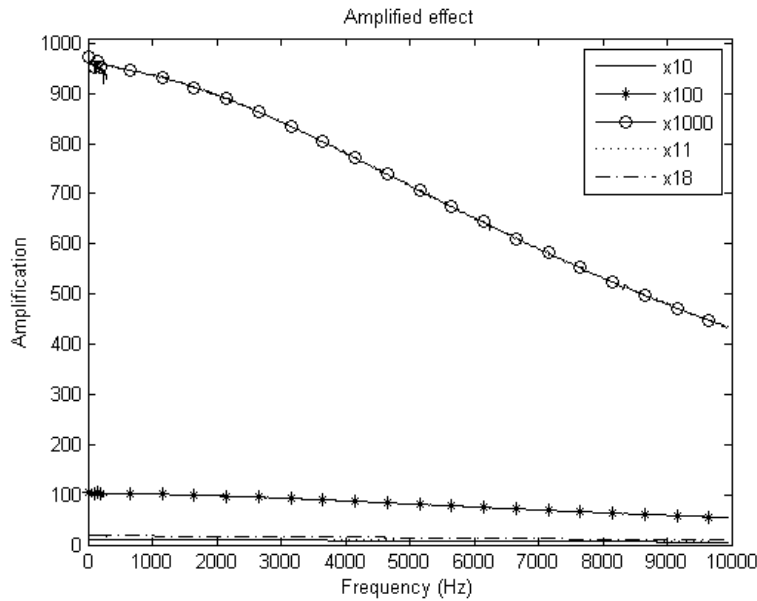


Figure 6.2: Comparison of the first 60 data points of the original sine signal obtained from the amplifier test.



(a) The amplified effect for the 5 levels of amplification tested for frequencies 1-300 Hz.



(b) The amplified effect for all the amplifications tested. 1-300 Hz and 150-10000 Hz plotted together.

Figure 6.3: The amplified effect for the frequencies 1-300 Hz (top), and the amplified effect for all the amplifications tested where 1-300 Hz and 150-10000 Hz are plotted together (bottom).

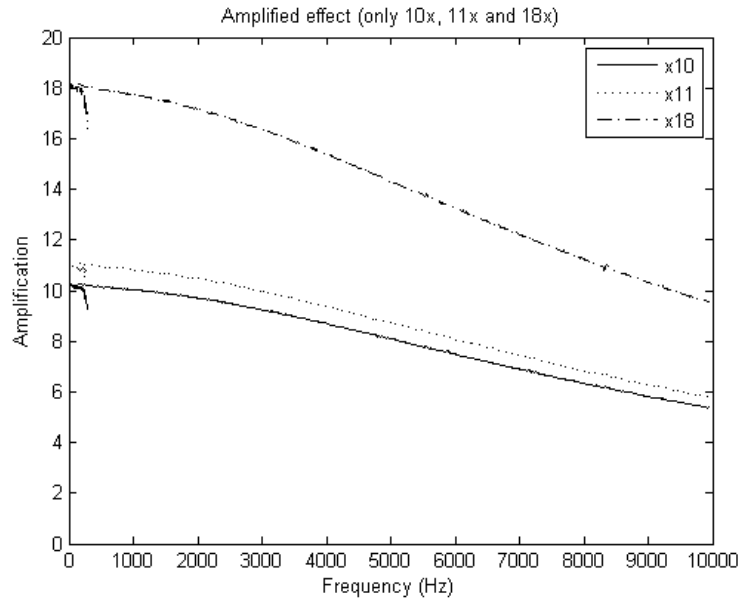


Figure 6.4: The amplified effect for the lower 3 levels of amplification.

Figure 6.3 shows the results of the amplifier tests. In figure 6.3a only the test for the frequencies 1-300 Hz is plotted. Figure 6.3b shows the amplified effect for all the frequencies that were tested. Since there is a wide range of amplifications, the lower frequencies may just seem flat because of the axis, so the three lower amplifications is therefore shown in figure 6.4 to get a better look at these amplifications. As seen in figure 6.3b the decrease for each of the amplifications seems to be less than half the value at 1 Hz. This corresponds to a fall of less than 3 dB, and therefore the amplifier can be qualified as linear.

Figure 6.5 shows a plot to illustrate the area where the two tests overlap. As seen in the figure, the amplified effect from 1-300 Hz decreases as the frequency increases, which also can be seen in figure 6.3b and 6.4. However, from 150 Hz, the amplified effect calculated from the test that was for 0-10000 Hz does not decrease, but is approximately constant. This may be due to how the sine wave generated for the test from 1-300 Hz was sampled. The last frequency, 300 Hz, was sampled with only 120 samples, giving only 8 samples per period (from equation 2.4 on page 8). While calculating the number of samples used for each frequency, some of the numbers are not integers. For the lower frequencies, this may not affect the sampled signal as much since the 15 periods are distributed over a large number of samples.



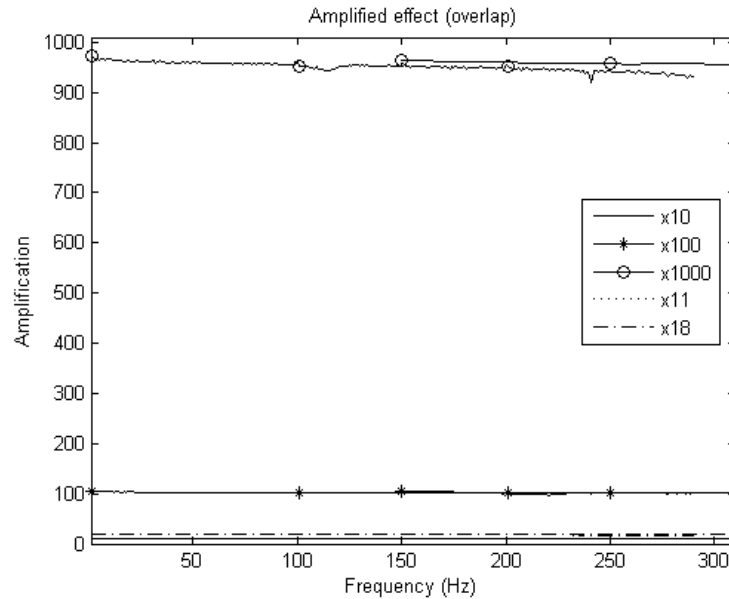


Figure 6.5: The amplified effect for all the amplifications tested. The plot is zoomed in on the frequencies where the two tests overlap.

For the highest frequencies however, the number of samples to represent one period is remarkably lower, and therefore it is possible that the real max and min point may have been missed due to the low number of samples. This may explain why the data from the 1-300 Hz decreases while the data from the other test stays approximately constant.

Figure 6.6 shows an earlier result of the amplifier test for 1-300 Hz. The amplified effect is approximately linear, except for the unexpected decrease and increase around 200 Hz for 1000 times amplification. Having no other logical explanation for it than it being an effect of external noise, the amplifier test in LabView was run once more with the 1000 times amplification. The new data set for 1000 times amplification is the one used in figure 6.3a. Due to the differences seen for the two data sets obtained for the 1000 times amplification, the decrease and increase is believed to be caused by external noise. This is something to have in mind when doing the experiments, that for example touching the cords connecting the amplifier may result in noise in the recorded signal.

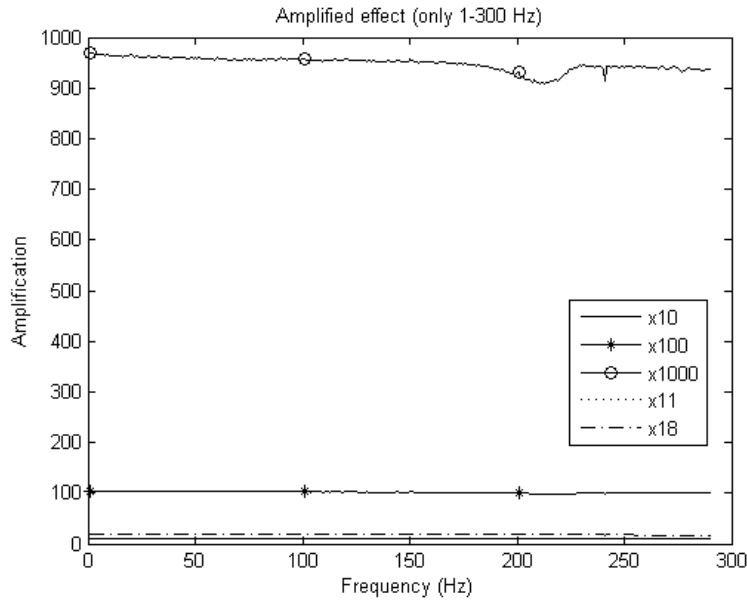
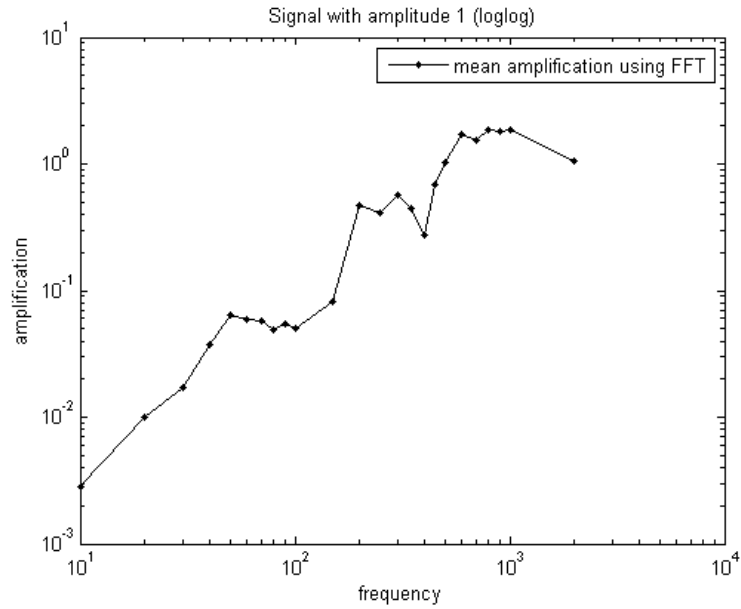


Figure 6.6: The amplified effect for the different amplifications for 1-300 Hz done in an earlier test of the amplifier.

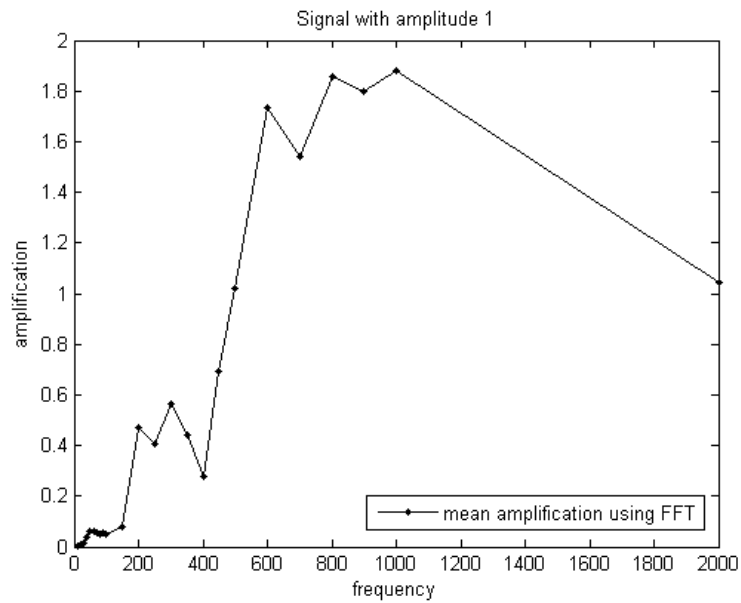
### 6.1.2 The microphone

Figure 6.7 shows the amplification of the different frequencies that were sent through the setup, both with non-logarithmic and logarithmic scales.

While running the program it was possible to hear the frequencies above 50 Hz from the headset that was used. When increasing the frequency, it also sounded like that the amplitude of the signal increased. This may have been a hearing deception since the frequency was increasing, but the decrease in amplitude around 400 Hz that can be seen in the figures could also be heard with the human ear. The increases and decreases of amplification may therefore have been the headset and not the microphone. Since it still is not possible to know what is caused by the headset and what is caused by the microphone, one can not with 100 % certainty know if the microphone is linear. However, it would seem that the microphone is able to record the frequencies, since it was possible to detect every frequency that was played.

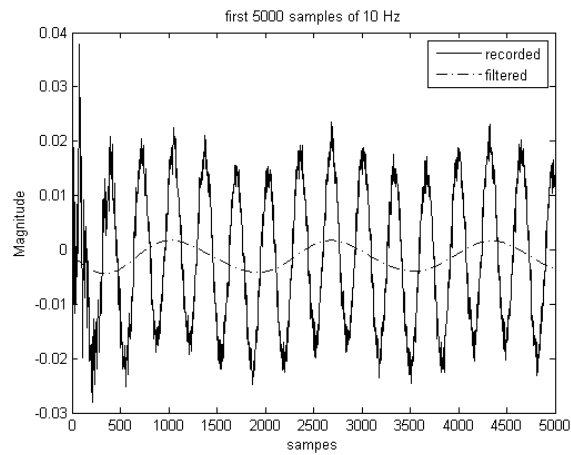


(a) The amplified effect for each frequency tested in the microphone test plotted in log-log.

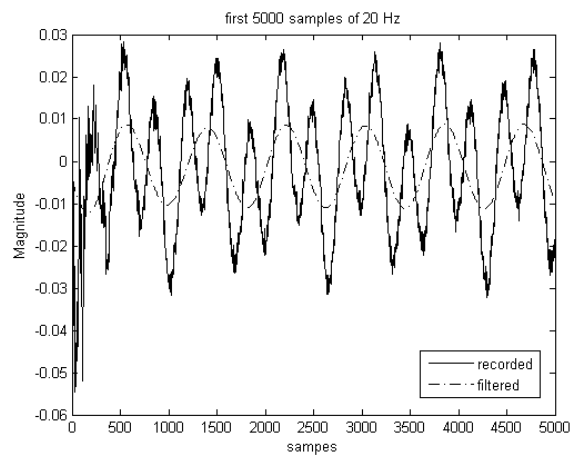


(b) The amplified effect for each frequency tested in the microphone test.

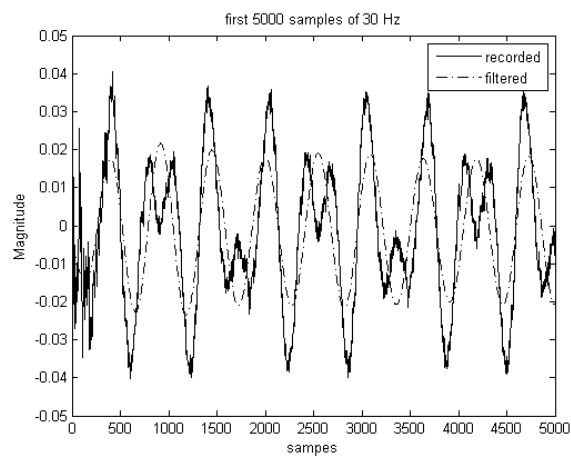
Figure 6.7: The amplified effect for each frequency that was tested. Plotted in both with logarithmic scales (top) and non-logarithmic scales (bottom).



(a) The recorded and filtered signal for 10 Hz plotted together.

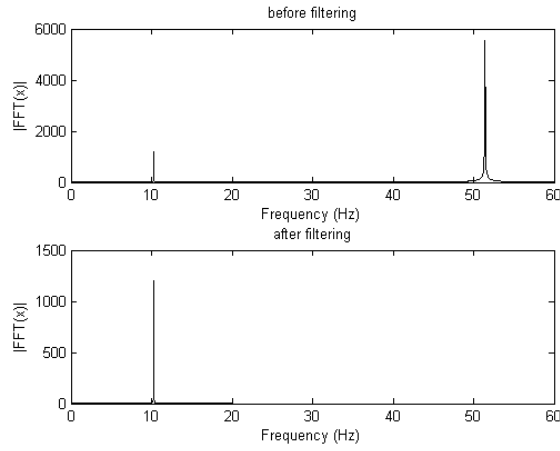


(b) The recorded and filtered signal for 20 Hz plotted together.

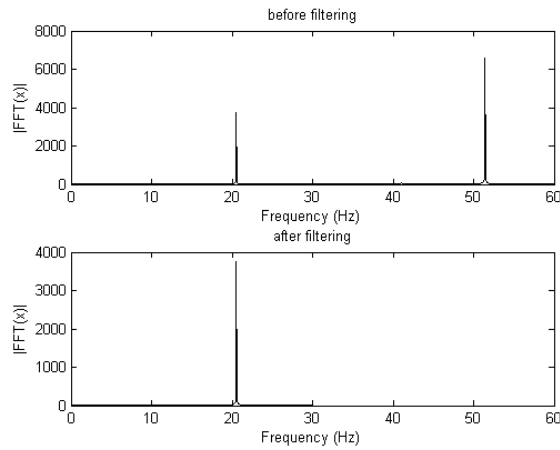


(c) The recorded and filtered signal for 30 Hz plotted together.

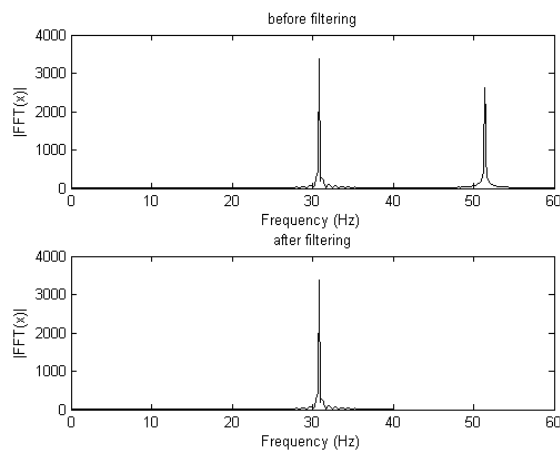
Figure 6.8: The recorded (raw) signal plotted together with the signal after filtering for the lowest frequencies.



(a) Fast Fourier Transform before and after filtering for 10 Hz.



(b) Fast Fourier Transform before and after filtering for 20 Hz.



(c) Fast Fourier Transform before and after filtering for 30 Hz.

Figure 6.9: Plot of the Fast Fourier Transform for the lowest frequencies before and after filtering to make sure that the wanted signal is not removed.

For the lowest frequencies seen in figure 6.7, the signals needed to be filtered before the amplification was calculated due to the large level of noise. Figure 6.8 shows the signal before and after the filtering. From the recorded signals in the figure, it is possible to see how the noise is affecting the original signal. From the figures it looks like the main frequency is, presumably, 50 Hz, but that the signal itself is oscillating in a lower frequency. From the filtered signals, it is possible to see that the extra fluctuations are gone, and what is left are the lower frequencies. To make sure that none of the desired frequencies were filtered, figure 6.9 shows the Fast Fourier transform of the signals before and after they were filtered. As seen in figure 6.9a and 6.9b the magnitude for the noise at 50 Hz is overpowering in comparison to the magnitudes at 10 and 20 Hz, respectively. In figure 6.9c, the magnitude at 30 Hz is larger than the magnitude at 50 Hz, but the signal to noise ratio is still very low. As seen in the after-plots, the 50 Hz noise is removed and the dominating frequency is the desired frequency.

For the 30 Hz, the two methods for finding the mean amplification was used, since the 30 Hz mostly had a larger amplitude than the noise. The two different values were printed out on the MATLAB screen, and the difference between the values are shown in the bottom of the code in appendix C.2.1. There's an approximately 12 % difference between the numbers, and it can be debated whether or not the amplification of this frequency is a bit higher than shown in figure 6.7. Seeing as the highest value (0.0195) was calculated through taking the mean of 15 FFTs of different signals, this value is probably the more realistic value. Substituting the value that is shown in figure 6.7 with this value results in the amplification from 10 to 50 Hz to have a more straight increase.

From the power spectrum shown in figure 1.1 from the introduction on page 3, there were frequencies detected that were under 10 Hz. Ideally these lower frequencies should also be tested such that the microphone qualification covered all the frequencies. However, due to the fact that a speaker needed to be used to qualify the microphone, there were some expected limitations for the lower frequencies. To be able to detect frequencies lower than 50 Hz was surprising, since there are very few speakers that can play such low frequencies. To detect frequencies lower than 10 Hz is assumed to be unrealistic, because of the low oscillation.

One thing that is certain from these results, is that it seems that the microphone was able to detect each frequency in the range that was tested. The uncertainty is whether or not the microphone is equally sensitive to each

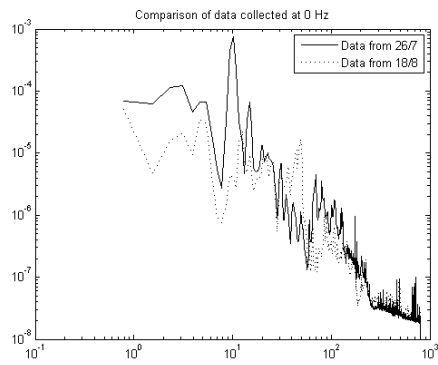
frequency. From what was experienced while running the test on the microphone, it would seem that most of the changes in the amplification came from the speaker, and that it is not unlikely that the microphone may in fact be equally sensitive to the frequencies. To investigate this further, the test could be run with different speakers to see if the results are different. Then by using the differences of the results from the tests, it may be possible to confirm that some of the changes in amplification was due to the speaker that was used. It is possible, however, that the type of microphone that was used was not the most ideal microphone, and for further research it could be a good idea to test out different microphones as well.

From the results from the tests run on the amplifier and microphone, the experimental setup has not been completely validated, because of the uncertainties when it comes to the microphone. However, with the extra tests mentioned above it may be possible to better distinguish between the microphone and speaker, and it is plausible that the whole experimental setup can be validated with the results from these extra tests.

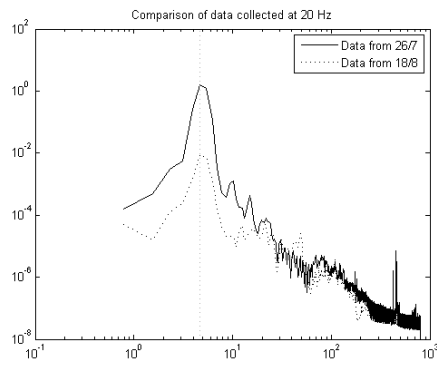
## 6.2 Results from the sound measurements

As mentioned in chapter 5, two sets of measurements were taken in the Hydrodynamic lab.

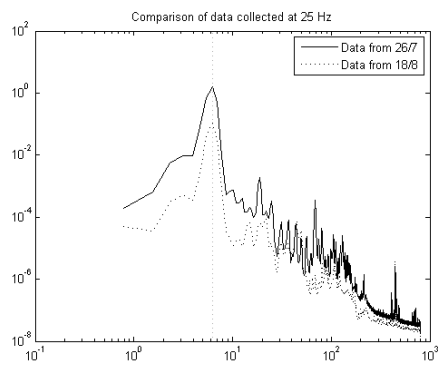
The first set of measurements contained fewer data points, and there were fewer measurements overall. Therefore the second set will be presented and discussed. However, the two sets were compared to see if there were any unexpected differences. Figure 6.10 shows the estimated power spectrums for the pump frequencies that were common for both sets. Figure 6.10a shows the estimated power spectrum for the noise (pump frequency 0 Hz), while figure 6.10b, 6.10c and 6.10d shows a comparison of the estimated power spectrums at different pump frequencies from the two sets. From the figure, it seems that the main difference is the magnitude. Since the two sets were taken weeks apart, it is plausible that this is due to unintentional change of settings on the amplifier. It is important, however, to note that the top point for the turbulent flows are located at the same point even though they have a different magnitude, and that the slope seems to be at the same angle.



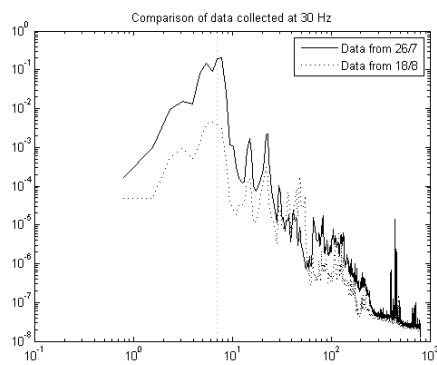
(a) Pump frequency 0 Hz.



(b) Pump frequency 20 Hz.



(c) Pump frequency 25 Hz.



(d) Pump frequency 30 Hz.

Figure 6.10: Comparison of the estimated power spectrum for the first and second set of measurements.



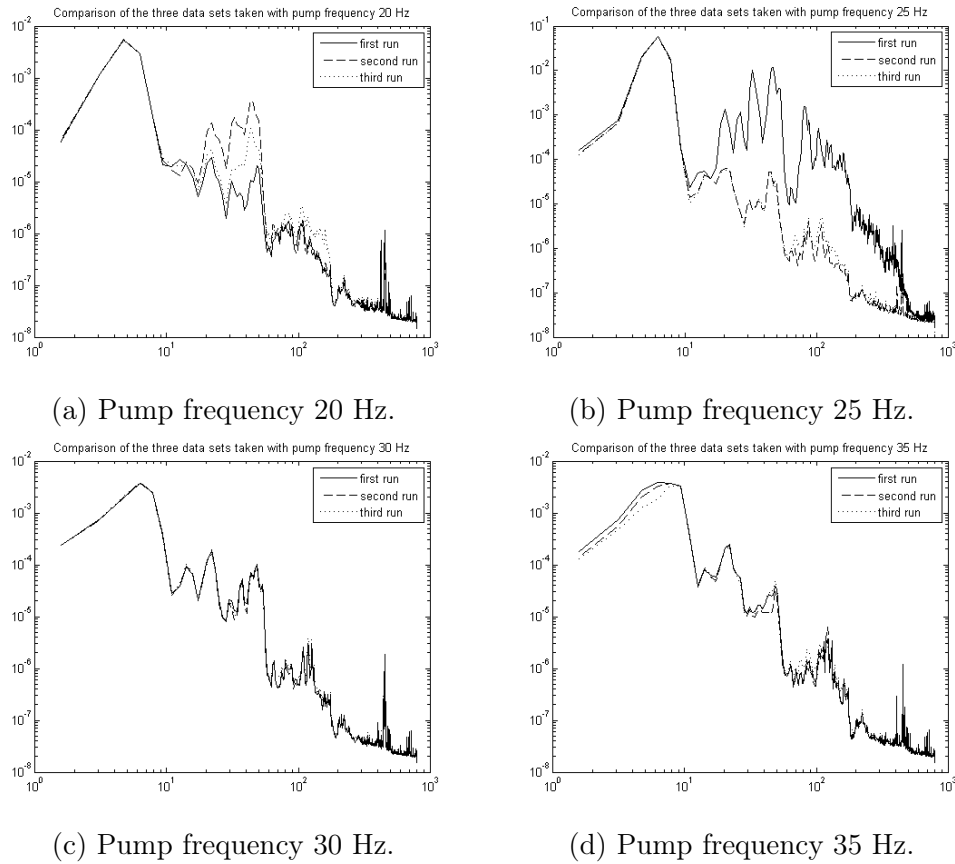


Figure 6.11: The power spectrum for the different data files for each pump frequency plotted together to check for reproducibility.

An own implementation was made for the estimating the power spectral density using the Welch's estimate due to uncertainties whether or not the built-in function in MATLAB normalized the data or how the normalization was done. For the first set of measurements, the home-made function for estimating the power spectral density and the built-in function in matlab were compared. It was discovered that the home-made function and the built-in function resulted in the same spectra, so the built-in function from MATLAB was used.

Figure 6.11 shows the different data files for the second set of measurements. The files containing data for the same pump frequency are plotted together to look for reproducibility. In addition to the data files that are plotted, there were 3 files taken with pump frequency 0 Hz, which should represent the background noise. In between some of the measurements there were

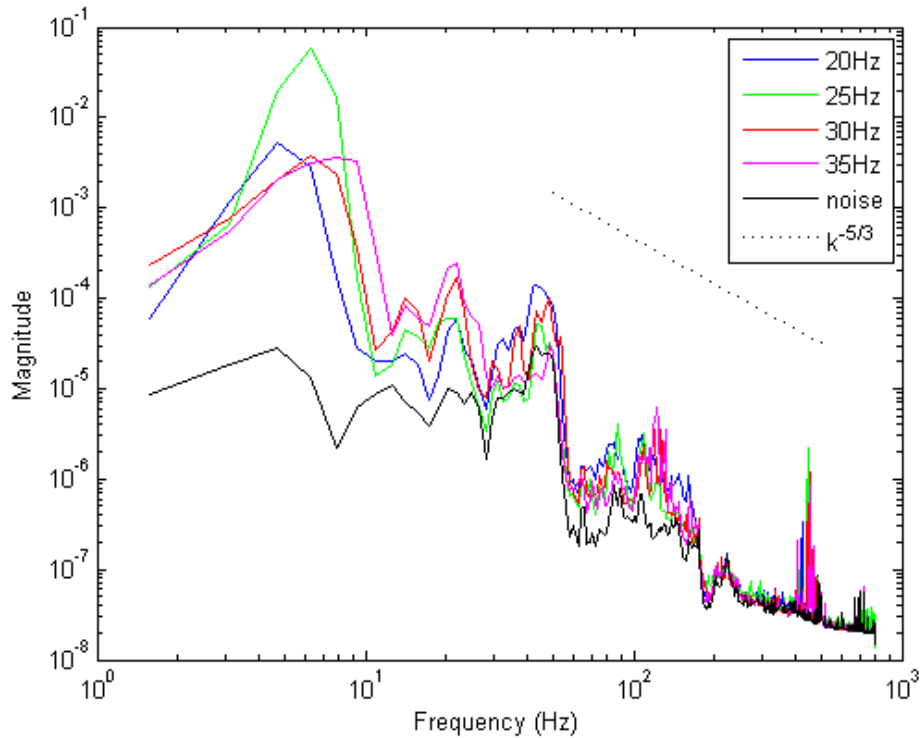


Figure 6.12: The estimated power spectrum for the second set of measurements, plotted together with the turbulent power profile.

maintanance work in one of the upper floors. The files that are plotted in the figure were not affected by this extra noise, but the maintainance work started during one of the measurements that was taken for the 25 Hz pump frequency. This file was discarded due to extensive background noise, and a new measurement was taken in its place. As seen in figure 6.11c and 6.11d the three measurements are almost identical. From figure 6.11b it would seem that the first run was affected by noise, since the two other measurements seem to be very similar. From figure 6.11a, it seems that there were some varying noise level between 20 and 60 Hz. Even though there are some differences between the results from the runs for each pump frequency, it would seem that the measuring technique is time-independent.

Figure 6.12 shows the estimated power spectrum for the second set of measurements, plotted together with the turbulent power profile. Since there were more than one measurement for each pump frequency, the measurement that seemed to have the least amount of noise was plotted. This was

done by visual inspection of the signals plotted as time series. The second run was used for 25 Hz and 35 Hz, and the third run was used for 20 Hz and 30 Hz. Shown by figure 6.11, it would seem that the measurements that were chosen were representative for their pump frequencies. However, in hindsight it would probably have been a better idea to use the first run for 20 Hz due to that it has the lowest amplitude in the area where the three runs are different.

As seen in the figure 6.12, the power spectrums for the turbulent flows follow the same fall and form as the power spectrum for the noise for especially the higher frequencies. However, from 1 to approximately 20 Hz the magnitudes for the turbulent flows are much higher than for the noise, and these values could come from real physics.

The slope that is seen at approximately 10 Hz is clearly a fall in energy, but compared to the energy profile, the fall should not be as steep as it is. This may be because the sound was recorded from the outside of the pipe. The sound propagates differently through water and through the pipe walls, and this can have affected the steepness of the power spectrum.

From the qualification of the microphone, it was not possible to be sure whether or not the results were from the microphone or the headset. If the headset was linear except from the sudden decreases that could be heard, it is possible that the slope of the estimated power spectrums should be steeper overall. In figure 6.12 the spectra seems to be flat at 10 to 50 Hz, but since figure 6.7a shows an increase of amplification in that area, it is possible that the spectra should be decreasing from 10 to 50 Hz. If the headset was the cause of the increase of magnification seen in figure 6.7a, it is plausible that the microphone is linear. If this is the case, a more extensive research needs to be conducted to determine more about the noise.

Since the power spectra does not completely follow the noise, it is also a possibility that the spectra would follow the energy profile if there was a way to filter out or compensate for the noise. From figure 6.11, and especially by looking at figure 6.11a and 6.11b, it would seem as if the additional background noise causes the deviation from the slope. If there is a default level of noise that is affecting every turbulent spectra, it is a possibility that the noise can be filtered out somehow by determining a mean power spectra of the noise by quantification. By a quick glance at figure 6.12 it would not seem impossible that the slope may follow the  $k^{-5/3}$ -rule if the power spectrum for the noise was just subtracted from the power spectra for each

turbulent flow.

To quantify the noise better, it may have been a good idea to have some measurements were the microphone just recorded the background noise without being attached to the pipe. Because the microphone was attached to the pipe, it is possible that some of the sounds that were recorded is due to the pump. Even though the measurements were taken sufficiently far away from the pump, it is not sure whether the pump noise propagated through the pipe and was recorded.

Even though there still is uncertainties when it comes to the steepness and the power spectrums compared to the energy profile for turbulent flows, one important thing to note is that the slopes for the different estimated power spectrums at different pump frequencies are placed as expected relative to their pump frequencies. From the theory, with increasing pump frequency, the power spectrum should shift to the right. Even though the qualification of the equipment did not have as pleasing results as desired, it would seem that detecting turbulence with microphone is difficult, but possible.

# Chapter 7

## Conclusion and further work

The results from the amplifier test shows that the drop for all amplifications is less than half of the initial values, in other words the drop is less than 3 dB, and from this the amplifier can be qualified as linear. However, the microphone could not be completely validated. It is not sure whether or not the microphone is equally sensitive to all frequencies, but from the results it is possible to confirm that the microphone can detect the frequencies in the range that was tested.

To be more certain of how the headset affected the results, it would be possible to run the microphone test again but with either a different speaker or microphone. By switching out only one of the two components at a time, it may be possible to use the results and their differences to distinguish between the microphone and the headset better. Using this method, it could also be possible to explore if there are better microphone options by only substituting the microphone.

For the turbulent spectra for the pipe flows, the spectra have a different slope than what would have been expected for turbulent flow. However, the spectra show some signs of being spectra of turbulent flow. Since there are similarities in the slope for the noise and the slope for the turbulent flows it is possible that further research can be done on the background noise to hopefully be able to filter the noise out or compensating for the noise in some way.

Even though there are uncertainties when it comes to the slopes of the tur-

bulent spectra, it is safe to say that some physics have been detected. By completing the qualification of the microphone by running the extra tests that are mentioned above, it should be possible to continue the steps towards the initial goal of this thesis, by adding a model of an aneurysm to the pipe loop, substituting parts of the pipe with elastic material, and experimenting with pulsative flows.

# Appendix A

## Settings for the measurement & Automation Explorer in LabView

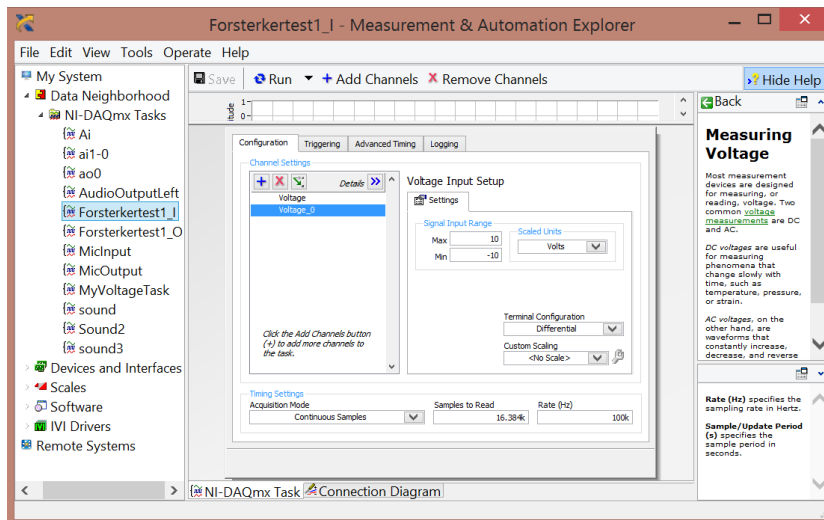
### A.1 Validation of the different components

#### A.1.1 Amplifier

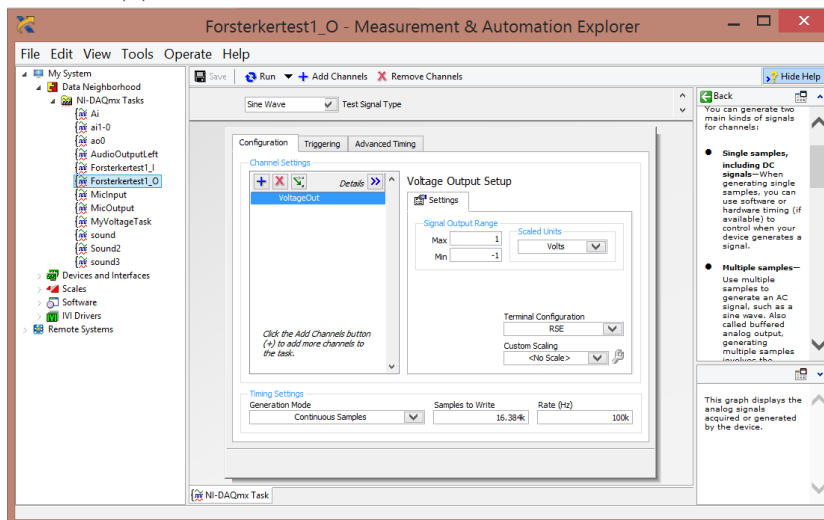
Figure A.1 shows the input and output settings that were used. The input, shown in A.1a, had two channels, the signal sent through the amplifier and the signal sent through just the NI myDAQ for delay determination, respectively. Other than the physical channels (from the wire connections on the NI myDAQ) there are no differences between the settings for the two input channels.

#### A.1.2 Microphone

Figure A.2 shows the settings for the input and output tasks that were used. The physical channel connected to the input task (shown in Figure A.2a) was "ai0", which is the orange wire from the amplifier, as seen in Figure 3.3 on page 15 (Ai 0+). The output task (shown in A.2b) was set to the noise cancelling headphone. The name of this task was a bit misleading, since the task actually has two physical channels, both the left and the right earpiece.



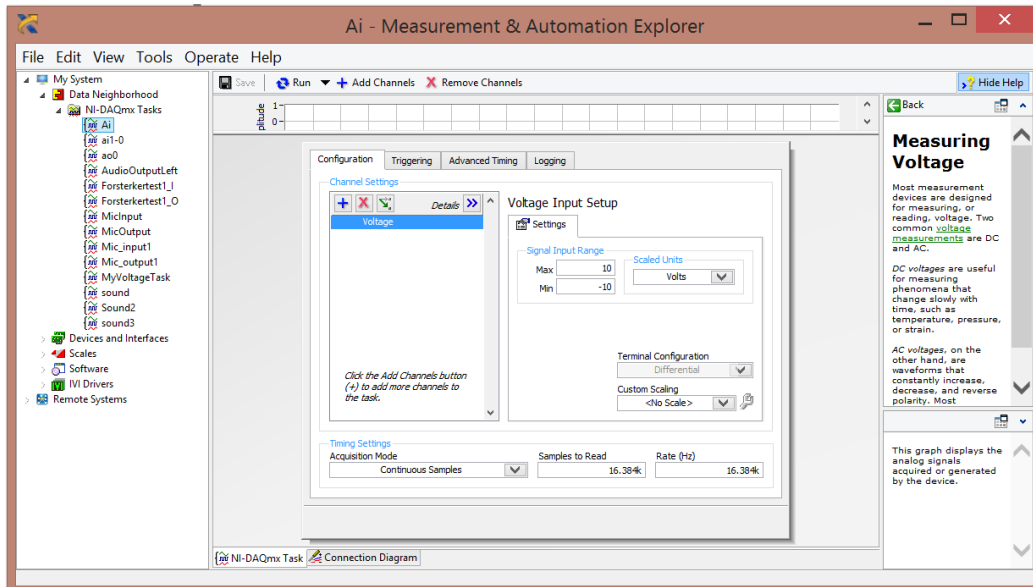
(a) Details for the input task for amplifier test for 150-10000 Hz.



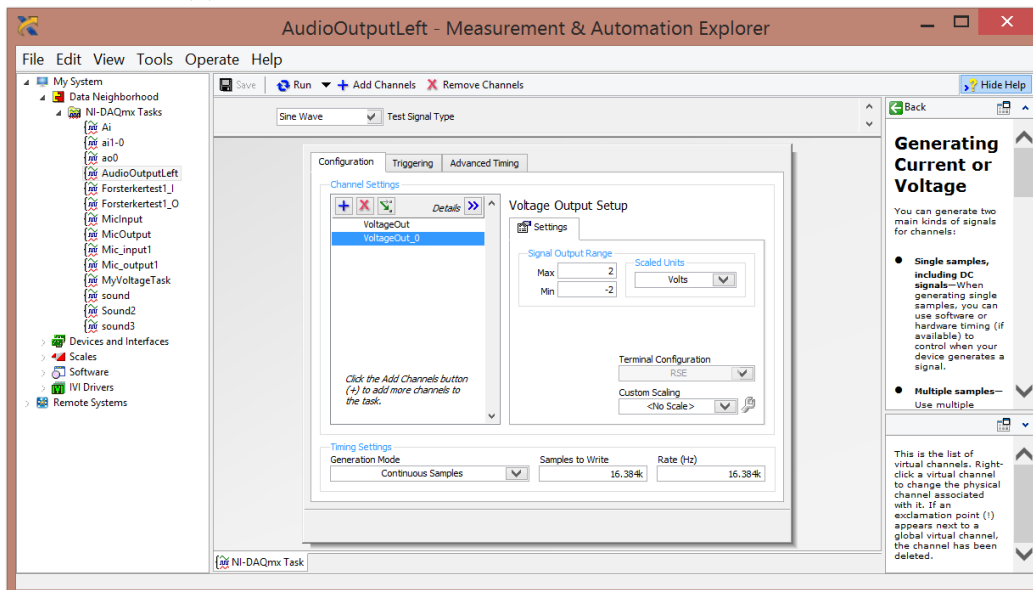
(b) Details for the output task for amplifier test for 150-10000 Hz.

Figure A.1: Input and output tasks for the amplifier test for 150-10000 Hz.





(a) Details for the input task for the microphone test.

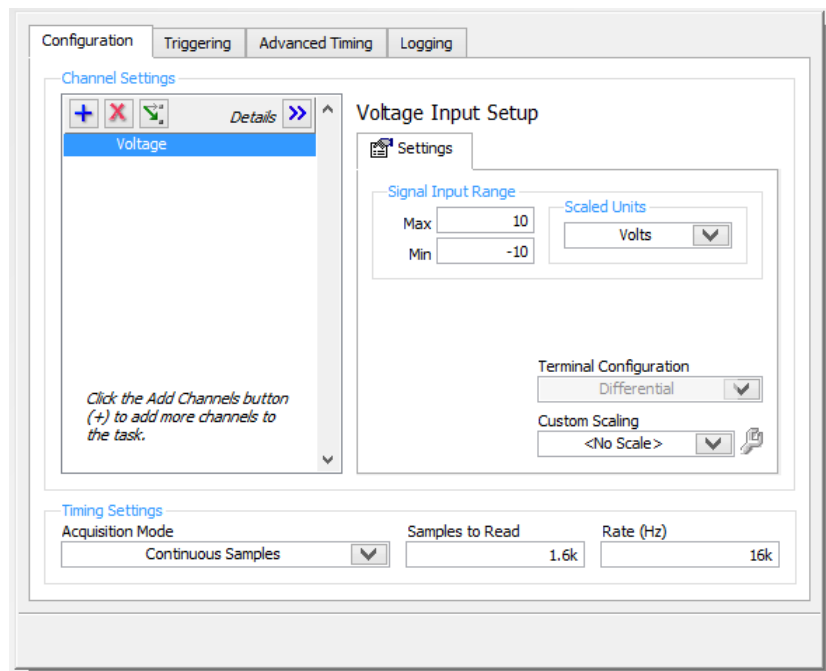


(b) Details for the output tasks for the microphone test.

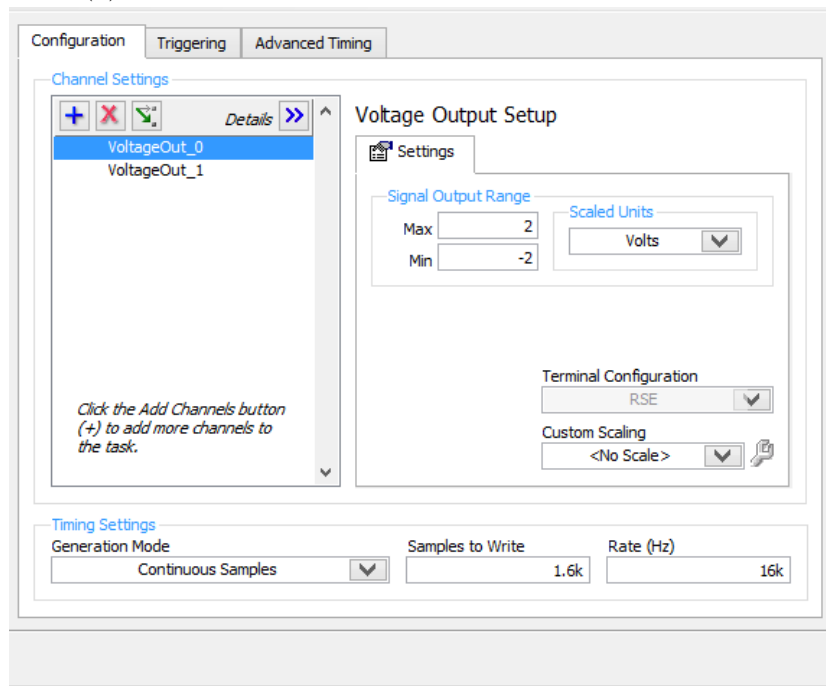
Figure A.2: The tasks settings for the microphone test.

## A.2 Sound measurements in turbulent pipe flow

Figure A.3 shows the input and output tasks for the LabView program used.



(a) Details for the input task for the pipe experiment.



(b) Details for the output task for the pipe experiment.

Figure A.3: Details for the input and output tasks for the LabView program used to record the pipe flows.



## Appendix B

### LabView program for sound measurements

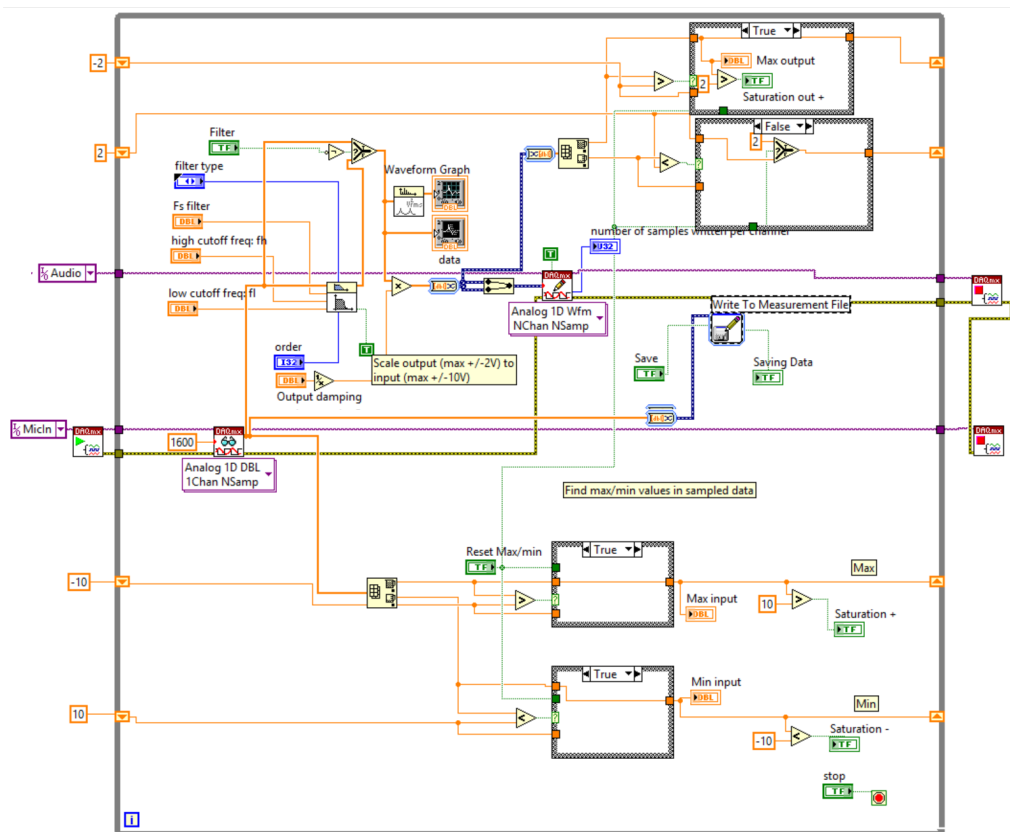


Figure B.1: Block diagram for the LabView program used to measure the pipe flows.

# Appendix C

## MATLAB files

### C.1 Qualification of the amplifier

#### C.1.1 amptest.m

---

```
1 clear all
2 close all
3
4 %% Importing data files for the 1-300 Hz test
5 datax10 = importdata('FTest2400_10x.lvm', '\t', 23);
6 datax100 = importdata('FTest2400_100x.lvm', '\t', 23);
7 datax1000 = importdata('FTest2400_1000x2.lvm', '\t', 23);
8 %datax1000 = importdata('FTest2400_1000x.lvm', '\t', 23);
9 datax11 = importdata('FTest2400_11x.lvm', '\t', 23);
10 datax18 = importdata('FTest2400_18x.lvm', '\t', 23);
11
12 %datax10 = lvm_import('forsterkertest2.lvm', '\t');
13 %%
14 x10 = datax10.data;
15 x100 = datax100.data;
16 x1000 = datax1000.data;
17 x11 = datax11.data;
18 x18 = datax18.data;
19
20 %% FOUND START POINTS AFTER ZOOMING IN ON THE DATASETS:
21 %The first approx 869 (x10: 863, x100 x1000: 869, x11 x18: 868,
```

```

22 %points are "noise", or an extra top in addition to the first
23 %15 periods we wanted, as you can see in the figure plotted below
24 %for x10:
25 figure ()
26 plot(0:length(x10(:,2))-1, x10(:,2), 'k')
27 xlim([0 48000])
28 title('The first 48000 points in x10')
29
30 %The total length of the array should be 226043
31 %(s = s+floor(36000/i)) for i=1:300
32 figure ()
33 subplot(2,1,1)
34 plot(0:length(x10(:,2))-1, x10(:,2), 'k')
35 xlim([0 2000])
36 title('Start and end of x10')
37 subplot(2,1,2)
38 plot(0:length(x10(:,2))-1, x10(:,3), 'k')
39 xlim([length(x10(:,2))-2000, length(x10(:,2))])
40
41
42 figure ()
43 subplot(2,1,1)
44 plot(0:length(x100(:,2))-1, x100(:,2))
45 xlim([0 2000])
46 title('Start and end of x100')
47 subplot(2,1,2)
48 plot(0:length(x100(:,2))-1, x100(:,3), 'g')
49 xlim([length(x100(:,2))-2000, length(x100(:,2))])
50
51
52 figure ()
53 subplot(2,1,1)
54 plot(0:length(x1000(:,2))-1, x1000(:,2))
55 xlim([0 2000])
56 title('Start and end of x1000')
57 subplot(2,1,2)
58 plot(0:length(x1000(:,2))-1, x1000(:,3), 'g')
59 xlim([length(x1000(:,2))-2000, length(x1000(:,2))])
60
61 figure ()
62 subplot(2,1,1)
63 plot(0:length(x11(:,2))-1, x11(:,2))
64 xlim([0 2000])

```



```

65 title('Start and end of x11')
66 subplot(2,1,2)
67 plot(0:length(x11(:,2))-1, x11(:,3), 'g')
68 xlim([length(x11(:,2))-2000, length(x11(:,2))])
69
70 figure()
71 subplot(2,1,1)
72 plot(0:length(x18(:,2))-1, x18(:,2))
73 xlim([0 2000])
74 title('Start and end of x18')
75 subplot(2,1,2)
76 plot(0:length(x18(:,2))-1, x18(:,3), 'g')
77 xlim([length(x18(:,2))-2000, length(x18(:,2))])
78
79 %% Removing noise to obtain the signal with 15 periods per frequency:
80 X10 = x10(941:end-970, 2);
81 X10_o = x10(941:end-970, 3);
82 t10 = 0:length(X10)-1;
83
84 figure()
85 subplot(2,1,1)
86 plot(0:length(X10)-1, X10, 'k')
87 title('Start and end of x10 after removal of noise')
88 xlim([0 2000])
89 subplot(2,1,2)
90 plot(0:length(X10)-1, X10, 'k')
91 xlim([length(X10)-50, length(X10)])
92
93 X100 = x100(946:end-965, 2); %870
94 X100_o = x100(946:end-965, 3);
95 t100 = 0:length(X100)-1;
96
97 figure()
98 subplot(2,1,1)
99 plot(0:length(X100)-1, X100)
100 title('Start and end of x100 after removal of noise')
101 xlim([0 2000])
102 subplot(2,1,2)
103 plot(0:length(X100)-1, X100)
104 xlim([length(X100)-50, length(X100)])
105
106 X1000 = x1000(947:end-964, 2); %859
107 X1000_o = x1000(947:end-964, 3);

```

```

108 t1000 = 0:length(X1000)-1;
109
110 figure()
111 subplot(2,1,1)
112 plot(0:length(X1000)-1, X1000)
113 title('Start and end of x1000 after removal of noise')
114 xlim([0 2000])
115 subplot(2,1,2)
116 plot(0:length(X1000)-1,X1000)
117 xlim([length(X1000)-50, length(X1000)])
118
119 X11 = x11(945:end-966, 2); %858
120 X11_o = x11(945:end-966, 3);
121 t11 = 0:length(X11)-1;
122
123 figure()
124 subplot(2,1,1)
125 plot(0:length(X11)-1, X11)
126 title('Start and end of x11 after removal of noise')
127 xlim([0 2000])
128 subplot(2,1,2)
129 plot(0:length(X11)-1,X11)
130 xlim([length(X11)-50, length(X11)])
131
132 X18 = x18(948:end-963, 2); %883
133 X18_o = x18(948:end-963, 3);
134 t18 = 0:length(X18)-1;
135
136 figure()
137 subplot(2,1,1)
138 plot(0:length(X18)-1, X18)
139 title('Start and end of x18 after removal of noise')
140 xlim([0 2000])
141 subplot(2,1,2)
142 plot(0:length(X18)-1,X18)
143 xlim([length(X18)-50, length(X18)])
144
145 %% Compare original sines
146 figure()
147 %Setting the plotcolors:
148 plot(-10,-10, 'k-+');
149 hold on;
150 plot(-10,-10, 'k-*');

```

```

151 plot(-10,-10, 'k-o');
152 plot(-10,-10, 'k-s');
153 plot(-10,-10, 'k-d');
154 legend('x10', 'x100', 'x1000', 'x11', 'x18');
155 %plot(t10, X10_o, 'k-')
156 plot(t10(1:200:end), X10_o(1:200:end), 'k-+')
157 hold on
158 %plot(t18, X18_o, 'k-')
159 plot(t18(1:200:end), X18_o(1:200:end), 'k-d')
160 %plot(t100, X100_o, 'k-')
161 plot(t100(1:200:end), X100_o(1:200:end), 'k-*')
162 %plot(t1000, X1000_o, 'k-')
163 plot(t1000(1:200:end), X1000_o(1:200:end), 'k-o')
164 %plot(t11, X11_o, 'k:')
165 plot(t11(1:200:end), X11_o(1:200:end), 'k-s')
166 xlim([0 10000])
167 text = 'Comparison of the original sine obtained. ';
168 text2 = 'Plotting every 200th point.';
169 title([text, text2])
170
171
172 %% compare the amplified
173 figure()
174 plot(t1000, X1000, 'r')
175 hold on
176 plot(t100, X100, 'g')
177 plot(t18, X18, 'm')
178 plot(t11, X11, 'y')
179 plot(t10, X10)
180 title('Comparison of the amplified');
181 legend('x1000', 'x100', 'x18', 'x11', 'x10');
182 %
183 arraylengths = [length(X10); length(X100); length(X1000);...
184 length(X11); length(X18)]
185 %
186
187 %% FINDING AMPLIFIED 1-300:
188 [amp10 amp100, amp1000, amp11, amp18] = ...
189 amp(X10, X100, X1000, X11, X18);
190
191 %% FINDING AMPLIFIED 150-10000:
192 [Amp10 Amp100, Amp1000, Amp11, Amp18] = datafull();
193

```

```

194 %% Plot of the amplified effect for 1-300 Hz:
195 figure()
196 %Setting the plotcolors and legend:
197 plot(-10,-10, 'k-');
198 hold on;
199 plot(-10,-10, 'k-*');
200 plot(-10,-10, 'k-o');
201 plot(-10,-10, 'k:');
202 plot(-10,-10, 'k-.');
203 legend('x10', 'x100', 'x1000', 'x11', 'x18');
204
205 plot(amp10, 'k');
206 plot(amp100, 'k');
207 plot(1:100:300, amp100(1:100:end), 'k*');
208 plot(amp1000, 'k');
209 plot(1:100:300, amp1000(1:100:end), 'ko');
210 plot(amp11, 'k:');
211 plot(amp18, 'k-.');
212 xlim([0 300])
213 title('Amplified effect (only 1-300 Hz)')
214 ylabel('Amplification')
215 xlabel('Frequency (Hz)')
216
217 %% Plot of the amplified effect all amplifications:
218 figure()
219 %Setting the plotcolors:
220 plot(-10,-10, 'k-');
221 hold on;
222 plot(-10,-10, 'k-*');
223 plot(-10,-10, 'k-o');
224 plot(-10,-10, 'k:');
225 plot(-10,-10, 'k-.');
226 legend('x10', 'x100', 'x1000', 'x11', 'x18');
227 plot(amp10, 'k');
228 plot(amp100, 'k');
229 plot(1:100:300, amp100(1:100:end), 'k*');
230 plot(amp1000, 'k');
231 plot(1:100:300, amp1000(1:100:end), 'ko');
232 plot(amp11, 'k:');
233 plot(amp18, 'k-.');
234 plot(150:10:9950, Amp10, 'k-');
235 plot(150:10:9950, Amp100, 'k-');
236 plot(150:500:9950, Amp100(1:50:end), 'k*');

```

```

237 plot(150:10:9950, Amp1000, 'k-');
238 plot(150:500:9950, Amp1000(1:50:end), 'ko');
239 plot(150:10:9950, Amp11, 'k:');
240 plot(150:10:9950, Amp18, 'k-.');
241 xlim([0 10000]);
242 ylim([0 1010]);
243 title('Amplified effect')
244 ylabel('Amplification')
245 xlabel('Frequency (Hz)')
246
247 %% Plot of the amplified effect for only 10x, 11x and 18x
248 figure()
249 %Setting the plotcolors:
250 plot(-10,-10, 'k-');
251 hold on;
252 plot(-10,-10, 'k:');
253 plot(-10,-10, 'k-.');
254 legend('x10', 'x11', 'x18');
255 plot(amp18, 'k-.')
256 hold on
257 plot(amp11, 'k:')
258 plot(amp10, 'k-')
259 ylim([0 20])
260 title('Amplified effect (only 10x, 11x and 18x)')
261 ylabel('Amplification')
262 xlabel('Frequency (Hz)')
263 %plot(150:10:9950, Amp10, 'k-');
264 %plot(150:10:9950, Amp11, 'k:');
265 %plot(150:10:9950, Amp18, 'k-.');
266
267 %%Plot of the overlap between the two tests:
268 figure()
269 %Setting the plotcolors:
270 plot(-10,-10, 'k-');
271 hold on;
272 plot(-10,-10, 'k-*');
273 plot(-10,-10, 'k-o');
274 plot(-10,-10, 'k:');
275 plot(-10,-10, 'k-.');
276 legend('x10', 'x100', 'x1000', 'x11', 'x18');
277 plot(amp10, 'k');
278 plot(amp100, 'k');
279 plot(1:100:300, amp100(1:100:end), 'k*');

```

```

280 plot(amp1000, 'k');
281 plot(1:100:300, amp1000(1:100:end), 'ko');
282 plot(amp11, 'k:');
283 plot(amp18, 'k-.');
284 plot(150:10:9950, Amp10, 'k-');
285 plot(150:10:9950, Amp100, 'k-');
286 plot(150:100:9950, Amp100(1:10:end), 'k*');
287 plot(150:10:9950, Amp1000, 'k-');
288 plot(150:100:9950, Amp1000(1:10:end), 'ko');
289 plot(150:10:9950, Amp11, 'k:');
290 plot(150:10:9950, Amp18, 'k-.');
291 xlim([1 310]);
292 ylim([0 1010]);
293 title('Amplified effect (overlap)')
294 ylabel('Amplification')
295 xlabel('Frequency (Hz)')
296
297 %% Plot of the amplified effect all amplifications in dB:
298 figure()
299 %Setting the plotcolors:
300 plot(-10,-10, 'k-');
301 hold on;
302 plot(-10,-10, 'k-*');
303 plot(-10,-10, 'k-o');
304 plot(-10,-10, 'k:');
305 plot(-10,-10, 'k-.');
306 legend('x10', 'x100', 'x1000', 'x11', 'x18');
307 plot(10*log10(amp10), 'k');
308 plot(10*log10(amp100), 'k');
309 plot(1:100:300, 10*log10(amp100(1:100:end)), 'k*');
310 plot(10*log10(amp1000), 'k');
311 plot(1:100:300, 10*log10(amp1000(1:100:end)), 'ko');
312 plot(10*log10(amp11), 'k:');
313 plot(10*log10(amp18), 'k-.');
314 plot(150:10:9950, 10*log10(Amp10), 'k-');
315 plot(150:10:9950, 10*log10(Amp100), 'k-');
316 plot(150:500:9950, 10*log10(Amp100(1:50:end)), 'k*');
317 plot(150:10:9950, 10*log10(Amp1000), 'k-');
318 plot(150:500:9950, 10*log10(Amp1000(1:50:end)), 'ko');
319 plot(150:10:9950, 10*log10(Amp11), 'k:');
320 plot(150:10:9950, 10*log10(Amp18), 'k-.');
321 xlim([0 10000]);
322 ylim([0 1010]);

```

```

323 title('Amplified effect in dB')
324 ylabel('Amplification (dB)')
325 xlabel('Frequency (Hz)')

```

---

### C.1.2 datafull.m

---

```

1 function [X10, X100, X1000, X11, X18] = datafull()
2 x10 = importdata('150-10000_omkjoring/forsterkertest__10x.lvm', ...
3     '\t', 23);
4 x100 = importdata('150-10000_omkjoring/forsterkertest__100x.lvm', ...
5     '\t', 23);
6 x1000 = importdata('150-10000_omkjoring/forsterkertest__1000x.lvm', ...
7     '\t', 23);
8 x11 = importdata('150-10000_omkjoring/forsterkertest__11x.lvm', ...
9     '\t', 23);
10 x18 = importdata('150-10000_omkjoring/forsterkertest__18x.lvm', ...
11     '\t', 23);
12
13 full10 = x10.data;
14 full100 = x100.data;
15 full1000 = x1000.data;
16 full11 = x11.data;
17 full18 = x18.data;
18
19 Full10 = full10(3109:end,2);
20 Full100 = full100(3109:end,2);
21 Full1000 = full1000(3109:end,2);
22 Full11 = full11(3109:end,2);
23 Full18 = full18(3109:end,2);
24
25 nrsamples = 10000;
26 startfr=0; %In reality this is 150 Hz
27 nrfreq = 980; %In reality this is 9800 Hz
28 amp10x = zeros(nrfreq-startfr,1);
29 amp100x = zeros(nrfreq-startfr,1);
30 amp1000x = zeros(nrfreq-startfr,1);
31 amp11x = zeros(nrfreq-startfr,1);
32 amp18x = zeros(nrfreq-startfr,1);
33
34 for freq=startfr:nrfreq
35     onepperiod = (nrsamples/(freq+1));

```

```

36     nrofperiods = 15;
37     tmpamp10 = zeros(15,1);
38     tmpamp100 = zeros(15,1);
39     tmpamp1000 = zeros(15,1);
40     tmpamp11 = zeros(15,1);
41     tmpamp18 = zeros(15,1);
42     %Finding start point for each frequency:
43     start = (freq*10000)+1;
44     for i=1:nrofperiods
45         slutt = start+ceil(oneperiod);
46         tmpamp10(i) = mean([max(Full10(start:slutt)), ...
47             max(-Full10(start:slutt))]);
48         tmpamp100(i) = mean([max(Full100(start:slutt)), ...
49             max(-Full100(start:slutt))]);
50         tmpamp1000(i) = mean([max(Full1000(start:slutt)), ...
51             max(-Full1000(start:slutt))]);
52         tmpamp11(i) = mean([max(Full11(start:slutt)), ...
53             max(-Full11(start:slutt))]);
54         tmpamp18(i) = mean([max(Full18(start:slutt)), ...
55             max(-Full18(start:slutt))]);
56         start = start + floor(oneperiod);
57     end
58     %Calculating mean amplifications:
59     amp10x(freq-(startfr-1)) = mean(tmpamp10);
60     amp100x(freq-(startfr-1)) = mean(tmpamp100);
61     amp1000x(freq-(startfr-1)) = mean(tmpamp1000);
62     amp11x(freq-(startfr-1)) = mean(tmpamp11);
63     amp18x(freq-(startfr-1)) = mean(tmpamp18);
64 end
65 %The original sine had amplitude 0.01, so multiplying with 100
66 %to get the real amplification:
67 X10 = amp10x*100;
68 X100 = amp100x*100;
69 X1000 = amp1000x*100;
70 X11 = amp11x*100;
71 X18 = amp18x*100;
72 end

```

---

### C.1.3 amp.m

---

```

1 function [Amp10, Amp100, Amp1000, Amp11, Amp18] = ...

```



```

2     amp(X10, X100, X1000, X11, X18)
3     nrfreq = 290;
4     amp1000 = zeros(nrfreq,1);
5     amp100 = zeros(nrfreq,1);
6     amp18 = zeros(nrfreq,1);
7     amp11 = zeros(nrfreq,1);
8     amp10 = zeros(nrfreq,1);
9
10    %%%
11    f1 = 36000;
12    period = 2400;
13    s = 0;
14    for i=1:nrfreq
15        in = s+1;
16        %Updating the current period sample length:
17        oneperiod = ((period)/(i));
18        %Making empty vectors to put the max/min-values in
19        locmax10 = zeros(30, 1);
20        locmax100 = zeros(30, 1);
21        locmax1000 = zeros(30, 1);
22        locmax11 = zeros(30, 1);
23        locmax18 = zeros(30, 1);
24        for j=1:15
25            %Start and endpoint of the period:
26            start = in;
27            slutt = in+ceil(oneperiod)-1;
28            %In addition to adding the max-point for each period,
29            %the -min is also added since it is also a "max-point":
30            locmax10(j) = max(X10(start:slutt))*100;
31            locmax10(j+15) = max(-X10(start:slutt))*100;
32            locmax100(j) = max(X100(start:slutt))*100;
33            locmax100(j+15) = max(-X100(start:slutt))*100;
34            locmax1000(j) = max(X1000(start:slutt))*100;
35            locmax1000(j+15) = max(-X1000(start:slutt))*100;
36            locmax11(j) = max(X11(start:slutt))*100;
37            locmax11(j + 15) = max(-X11(start:slutt))*100;
38            locmax18(j) = max(X18(start:slutt))*100;
39            locmax18(j+15) = max(-X18(start:slutt))*100;
40
41            %New start point is calculated
42            in = in+floor(oneperiod);
43        end
44

```

```

45     s = s + f1;
46     amp10(i) = mean(locmax10);
47     amp100(i) = mean(locmax100);
48
49     amp1000(i) = mean(locmax1000);
50     amp11(i) = mean(locmax11);
51     amp18(i) = mean(locmax18);
52     f1 = floor(36000/(i+1));
53 end
54 Amp10 = amp10;
55 Amp100 = amp100;
56 Amp1000 = amp1000;
57 Amp11 = amp11;
58 Amp18 = amp18;
59 end

```

---

## C.2 Qualification of the microphone

### C.2.1 mictest.m

---

```

1  clear all
2  close all
3
4  %% Importing data file
5  datafull = importdata('headphone_18.lvm', '\t', 22);
6
7  %Number of samples per run (point in datato1000);
8  tolength = 16384;
9
10 %datax10 = lvm_import('forsterkertest2.lvm', '\t');
11 %%
12 z = datafull.data;
13
14
15 %Frequencies goes from 20-100 with 10 step, 100-500
16 %with 50 step, 500-1000 with 100 step, and 1000-2000
17 %with 1000 step
18
19 t2 = 30:10:90;

```

```

20 t = 100:50:500;
21 t1 = 600:100:1000;
22
23 %Array with all the possible frequencies.
24 fullt = [t2 t t1 2000];
25
26 %max(FFT) = 0.5*amplitude
27
28 meantoamp = zeros(length(fullt),1);
29
30 counteroffreq = zeros(length(fullt),1);
31 freq30 = 0;
32
33 %Flipping such that the array has the same order as the
34 %measured files (biggest frequency first)
35 fullt = fliplr(fullt);
36
37 tmpcn = 1;
38 %Finding the mean for frequencies > 30
39 cc = gray(length(fullt));
40 figure()
41 for i=1:length(fullt)
42     tmp = zeros(1,1);
43     counter = 1;
44     for j=1:length(z(:,2))
45         if z(j,2) == fullt(i)
46             tmp(counter) = z(j,3);
47             counter=counter+1;
48             if j~=length(z(:,2))
49                 if (z(j+1,2) ~= fullt(i))
50                     break;
51                 end
52             end
53         end
54     end
55     tmpcn = tmpcn + counter;
56     meantoamp(i) = mean(tmp);
57     counteroffreq(i) = counter-1;
58     %Plotting to see the values in tmp to especially
59     %make sure that 50 Hz is correct.
60     plot(tmp, 'color', cc(i, :))
61     hold on
62 end

```

```

63
64 amp = (meantoamp.*2)/totlength;
65 sin102030 = fliplr(sin1030());
66
67 %Using the 30 Hz value that was found not by filtering:
68 totamp = [amp; sin102030(2:end)'];
69 figure()
70 fullx = [fullt 20 10];
71 plot(fullx , totamp, 'k.-')
72 legend('mean amplification using FFT')
73 xlabel('frequency')
74 ylabel('amplification')
75 title('Signal with amplitude 1')
76
77 %importing the sine obtained
78 sinus = importdata('signalobtain_2.lvm', '\t', 22);
79
80 figure()
81 loglog(fullx , totamp, 'k.-')
82 legend('mean amplification using FFT')
83 xlabel('frequency')
84 ylabel('amplification')
85 title('Signal with amplitude 1 (loglog)')
86
87 %30 Hz has been calculated through both the generated test and with
88 %filtering the sine signal. Printed out to screen for comparison:
89 amp30 = amp(end)
90 amp30filt = sin102030(1)
91
92
93 %% OUPUT:
94 % amp30 =
95 %      0.0195
96 % amp30filt =
97 %      0.0172

```

---

## C.2.2 sin1030.m

---

```

1 function [amp] = sin1030()
2 sinus10 = importdata('signal10.lvm', '\t', 22);
3 sinus20 = importdata('signal20.lvm', '\t', 22);

```

```

4  sinus30 = importdata('signal30.lvm', '\t', 22);
5
6  sin10 = sinus10.data;
7  sin20 = sinus20.data;
8  sin30 = sinus30.data;
9
10 figure()
11 plot(sin10(:,1), sin10(:,2));
12 hold on
13 plot(sin20(:,1), sin20(:,2), 'r');
14 plot(sin30(:,1), sin30(:,2), 'g');
15
16 Fs = 16834;
17 T = 1/Fs;
18 L1 = length(sin10(:,2));
19 L2 = length(sin20(:,2));
20 L3 = length(sin30(:,2));
21
22 NFFT1 = L1;
23 NFFT2 = L2;
24 NFFT3 = L3;
25
26 F1 = ((0:1/NFFT1:1-1/NFFT1)*Fs).';
27 F2 = ((0:1/NFFT2:1-1/NFFT2)*Fs).';
28 F3 = ((0:1/NFFT3:1-1/NFFT3)*Fs).';
29
30 %Finding the FFT
31 X1 = fft(sin10(:,2), NFFT1);
32 X2 = fft(sin20(:,2), NFFT2);
33 X3 = fft(sin30(:,2), NFFT3);
34
35 f1 = Fs/2*linspace(0,1, NFFT1/2+1);
36 f2 = Fs/2*linspace(0,1, NFFT2/2+1);
37 f3 = Fs/2*linspace(0,1, NFFT3/2+1);
38
39 figure()
40 plot(F1, abs(X1));
41
42 Y1 = X1;
43 Y2 = X2;
44 Y3 = X3;
45
46 [tmp1 index] = min(abs(F1-20));

```

```

47 Y1(index) = 0.5*Y1(index);
48 Y1(end-index+1) = 0.5*Y1(end-index+1);
49 Y1(index+1:end-index) = 0;
50
51 [tmp1 index] = min(abs(F2-30));
52 Y2(index) = 0.5*Y2(index);
53 Y2(end-index+1) = 0.5*Y2(end-index+1);
54 Y2(index+1:end-index) = 0;
55
56 [tmp1 index] = min(abs(F3-40));
57 Y3(index) = 0.5*Y3(index);
58 Y3(end-index+1) = 0.5*Y3(end-index+1);
59 Y3(index+1:end-index) = 0;
60
61 Z1 = ifft(Y1, 'symmetric');
62 Z2 = ifft(Y2, 'symmetric');
63 Z3 = ifft(Y3, 'symmetric');
64
65 T1 = fft(Z1, NFFT1);
66 T2 = fft(Z2, NFFT2);
67 T3 = fft(Z3, NFFT3);
68
69 t1max = max(2*abs(Y1(1:NFFT1/2+1)));
70 t2max = max(2*abs(Y2(1:NFFT2/2+1)));
71 t3max = max(2*abs(Y3(1:NFFT3/2+1)));
72
73 amp = [t1max/NFFT1, t2max/NFFT2, t3max/NFFT3];
74
75 figure()
76 plot(sin10(:,2), 'k');
77 hold on
78 plot((Z1), 'k-.')
79 xlim([0 5000])
80 title('first 5000 samples of 10 Hz')
81 legend('recorded', 'filtered');
82 xlabel('samps'); ylabel('Magnitude')
83
84 figure()
85 plot(sin20(:,2), 'k');
86 hold on
87 plot((Z2), 'k-.')
88 xlim([0 5000])
89 title('first 5000 samples of 20 Hz')

```

```

90 legend('recorded', 'filtered');
91 xlabel('samples'); ylabel('Magnitude')
92
93 figure()
94 plot(sin30(:,2), 'k');
95 hold on
96 plot((Z3), 'k-.');
97 xlim([0 5000])
98 title('first 5000 samples of 30 Hz')
99 legend('recorded', 'filtered');
100 xlabel('samples'); ylabel('Magnitude')
101
102 figure()
103 subplot(2,1,1)
104 plot(f1, 2*abs(X1(1:NFFT1/2+1)), 'k');
105 xlim([0 60])
106 xlabel('Frequency (Hz)'); ylabel('|FFT(x)|');
107 title('before filtering')
108 subplot(2,1,2)
109 plot(f1, 2*abs(T1(1:NFFT1/2+1)), 'k');
110 title('after filtering');
111 xlim([0 60])
112 xlabel('Frequency (Hz)'); ylabel('|FFT(x)|');
113
114 figure()
115 subplot(2,1,1)
116 plot(f2, 2*abs(X2(1:NFFT2/2+1)), 'k');
117 xlim([0 60])
118 title('before filtering')
119 xlabel('Frequency (Hz)'); ylabel('|FFT(x)|');
120 subplot(2,1,2)
121 plot(f2, 2*abs(T2(1:NFFT2/2+1)), 'k');
122 title('after filtering');
123 xlim([0 60])
124 xlabel('Frequency (Hz)'); ylabel('|FFT(x)|');
125
126 figure()
127 subplot(2,1,1)
128 plot(f3, 2*abs(X3(1:NFFT3/2+1)), 'k');
129 xlim([0 60])
130 xlabel('Frequency (Hz)'); ylabel('|FFT(x)|');
131 title('before filtering')
132 subplot(2,1,2)

```

```

133 plot(f3, 2*abs(T3(1:NFFT3/2+1)), 'k');
134 title('after filtering');
135 xlim([0 60])
136 xlabel('Frequency (Hz)'); ylabel('|FFT(x)|');
137 end

```

---

## C.3 Sound measurements

### C.3.1 welchpsd.m

---

```

1 function [Psd, F] = welchpsd(F_, A, fs)
2     N = length(A);
3     [Pxx, fm] = periodogram(A, hamming(N), [], fs);
4     L = length(F_*2)-1; %F1 is 'onesided'
5     D = L/2;
6     K = (N-L)/D+1;
7     wn = hamming(L);
8     n1 = 1;
9     n0=round(0.5*L);
10    Pw = 0;
11    w = fm; %In Hz
12
13    for i=1:K
14        wx = A(n1:(n1+L-1)).*wn;
15        Pw = Pw + (abs(fft(wx)).^2)/norm(wn);
16        n1 = n1+n0;
17    end
18    sl = round(length(F_)/2);
19    Psd = Pw(1:sl)/max(Pw);
20    F = F_(1:sl);
21 end

```

---

### C.3.2 normpsd.m

---

```

1 clear all
2 %close all
3

```



```

4  %% Importing data files :
5  f1 = 'TestMicInput_1.lvm';
6  f2 = 'TestMicInput_2.lvm';
7  f21 = 'TestMicInput_21.lvm';
8  f3 = 'TestMicInput_3.lvm';
9  f4 = 'TestMicInput_4.lvm';
10
11 lvf1 = importdata(f1, '\t', 22);
12 lvf2 = importdata(f2, '\t', 22);
13 lvf21 = importdata(f21, '\t', 22);
14 lvf3 = importdata(f3, '\t', 22);
15 lvf4 = importdata(f4, '\t', 22);
16
17 data1 = lvf1.data;
18 data2 = lvf2.data;
19 data21 = lvf21.data;
20 data3 = lvf3.data;
21 data4 = lvf4.data;
22
23 A1 = data1(:,2);
24 A2 = data2(:,2);
25 A21 = data21(:,2);
26 A3 = data3(:,2);
27 A4 = data4(:,2);
28
29 A21 = A21(1256001:end); %TestMicInput_2 has length 1256000.
30
31 Fs = 1/1600;
32 t1 = 1:length(A1);
33 t2 = 1:length(A2);
34 t21 = 1:length(A21);
35 t3 = 1:length(A3);
36 t4 = 1:length(A4);
37
38 t1 = t1*Fs;
39 t2 = t2*Fs;
40 t21 = t21*Fs;
41 t3 = t3*Fs;
42 t4 = t4*Fs;
43 %figure()
44
45 %%PLOTTING
46 figure()

```

```

47
48 WW = 1600;
49 fs = 1600;
50
51 [Psd1, F1] = pwelch(A1-mean(A1), WW, [], [], fs, 'onesided');
52 %loglog(F1, Psd1, 'b')
53 [Psd1, F1] = welchpsd(F1, A1, fs);
54 loglog(F1, Psd1, 'b')
55
56 hold on
57 [Psd2, F2] = pwelch(A2-mean(A2), WW, [], [], fs, 'onesided');
58 %loglog(F2, Psd2, 'r')
59 [Psd, F] = welchpsd(F2, A2, fs);
60 loglog(F2, Psd2, 'r')
61 %
62 [Psd21, F21] = pwelch(A21-mean(A21), WW, [], [], fs, 'onesided');
63 %loglog(F21, Psd21, 'g')
64 [Psd, F] = welchpsd(F21, A21, fs);
65 loglog(F21, Psd21, 'g')
66 %
67 [Psd3, F3] = pwelch(A3-mean(A3), WW, [], [], fs, 'onesided');
68 %loglog(F3, Psd3, 'm')
69 [Psd, F] = welchpsd(F3, A3, fs);
70 loglog(F3, Psd3, 'm')
71
72 [Psd4, F4] = pwelch(A4-mean(A4), WW, [], [], fs, 'onesided'); %noise
73 %loglog(F4, Psd4, 'c')
74
75 xlabel('Frequency (Hz)');
76 ylabel('Magnitude');
77 legend('31 Hz', '25 Hz', '20 Hz', '30 Hz', 'k^{-5/3}')
78
79 k = 50:1000;
80 plot(k, k.^(-5/3), 'k:');
81
82 %% Plotting MATLABs Psd against own implemented version:
83 % figure()
84 % loglog(F1, Psd1);
85 % hold on
86 % loglog(Ft, Psdt, 'r');

```

---

### C.3.3 nyedata.m

---

```
1 clear all
2 close all
3
4 %Importing files:
5 f1 = '140818tubeTest_3.lvm';
6 f2 = '140818tubeTest_6.lvm';
7 f3 = '140818tubeTest_10.lvm';
8 f4 = '140818tubeTest_12.lvm';
9 f5 = '140818tubeTest_16.lvm';
10
11 lvf1 = importdata(f1, '\t', 22);
12 lvf2 = importdata(f2, '\t', 22);
13 lvf3 = importdata(f3, '\t', 22);
14 lvf4 = importdata(f4, '\t', 22);
15 lvf5 = importdata(f5, '\t', 22);
16
17 data1 = lvf1.data;
18 data2 = lvf2.data;
19 data3 = lvf3.data;
20 data4 = lvf4.data;
21 data5 = lvf5.data;
22
23 A1 = data1(:,2);
24 A2 = data2(:,2);
25 A3 = data3(:,2);
26 A4 = data4(:,2);
27 A5 = data5(:,2);
28
29 %Finding the minimum length of the data sets:
30 arrlen = [length(A1), length(A2), ...
31           length(A3), length(A4), length(A5)];
32 test = min(arrlen);
33
34 %Finding the rms:
35 u1 = A1-mean(A1);
36 u2 = A2-mean(A2);
37 u3 = A3-mean(A3);
38 u4 = A4-mean(A4);
39 stoy = A5-mean(A5);
40
41 %resizing the data so they have the same length, so
```

```

42 %they can be compared.
43 A1 = u1(1:test);
44 A2 = u2(1:test);
45 A3 = u3(1:test);
46 A4 = u4(1:test);
47 A5 = stoy(1:test);
48
49 Fs = 1/16000;
50
51 %%FINDING THE PSDS AND PLOTTING
52 figure ()
53 WW = 850;
54 fs = 1600;
55 [Psd1, F1] = pwelch(A1, WW, [], [], fs, 'onesided');
56 loglog(F1, Psd1, 'b')
57 hold on
58
59 [Psd2, F2] = pwelch(A2, WW, [], [], fs, 'onesided');
60 loglog(F2, Psd2, 'g')
61
62 [Psd3, F3] = pwelch(A3, WW, [], [], fs, 'onesided');
63 loglog(F3, Psd3, 'r')
64
65 [Psd4, F4] = pwelch(A4, WW, [], [], fs, 'onesided');
66 loglog(F4, Psd4, 'm')
67
68 [Psd5, F5] = pwelch(A5, WW, [], [], fs, 'onesided');
69 loglog(F5, Psd5, 'k');
70
71 k = 50:500;
72
73 plot(k, k.^(-5/3), 'k:')
74 legend('20Hz', '25Hz', '30Hz', '35Hz', 'noise', 'k^{-5/3}')
75 xlabel('Frequency (Hz)')
76 ylabel('Magnitude')
77
78
79 %Just testing out something for fun:
80 % figure ()
81 % loglog(F1, abs(Psd1-Psd5), 'b');
82 % hold on
83 % loglog(F2, abs(Psd2-Psd5), 'r');
84 % loglog(F3, abs(Psd3-Psd5), 'g');

```

```

85 % loglog(F4, abs(Psd4-Psd5), 'm');
86 %
87 % loglog(F5, Psd5, 'k');
88 % k = 20:110;
89 % plot(k, k.^(-5/3), 'k:');
90 % k = 110:800;
91 % plot(k, 500*k.^(-3), 'k:');

```

---

### C.3.4 repetivity.m

---

```

1  clear all
2  close all
3
4  %Importing files:
5  f20_1 = importdata('140818tubeTest_1.lvm', '\t', 22);
6  f20_2 = importdata('140818tubeTest_2.lvm', '\t', 22);
7  f20_3 = importdata('140818tubeTest_3.lvm', '\t', 22);
8
9  f25_1 = importdata('140818tubeTest_4.lvm', '\t', 22);
10 f25_2 = importdata('140818tubeTest_6.lvm', '\t', 22);
11 f25_3 = importdata('140818tubeTest_7.lvm', '\t', 22);
12
13 f30_1 = importdata('140818tubeTest_8.lvm', '\t', 22);
14 f30_2 = importdata('140818tubeTest_9.lvm', '\t', 22);
15 f30_3 = importdata('140818tubeTest_10.lvm', '\t', 22);
16
17 f35_1 = importdata('140818tubeTest_11.lvm', '\t', 22);
18 f35_2 = importdata('140818tubeTest_12.lvm', '\t', 22);
19 f35_3 = importdata('140818tubeTest_13.lvm', '\t', 22);
20
21 f20_1 = f20_1.data(:,2);
22 f20_2 = f20_2.data(:,2);
23 f20_3 = f20_3.data(:,2);
24 f25_1 = f25_1.data(:,2);
25 f25_2 = f25_2.data(:,2);
26 f25_3 = f25_3.data(:,2);
27 f30_1 = f30_1.data(:,2);
28 f30_2 = f30_2.data(:,2);
29 f30_3 = f30_3.data(:,2);
30 f35_1 = f35_1.data(:,2);
31 f35_2 = f35_2.data(:,2);

```

```

32 f35_3 = f35_3.data(:,2);
33
34 %Finding the minimum length:
35 min20 = min([length(f20_1), length(f20_2), length(f20_3)]);
36 min25 = min([length(f25_1), length(f25_2), length(f25_3)]);
37 min30 = min([length(f30_1), length(f30_2), length(f30_3)]);
38 min35 = min([length(f35_1), length(f35_2), length(f35_3)]);
39
40 minlength = min([min20, min25, min30, min35]);
41
42 %Finding the rms:
43 u20_1 = f20_1 - mean(f20_1);
44 u20_2 = f20_2 - mean(f20_2);
45 u20_3 = f20_3 - mean(f20_3);
46 u25_1 = f25_1 - mean(f25_1);
47 u25_2 = f25_2 - mean(f25_2);
48 u25_3 = f25_3 - mean(f25_3);
49 u30_1 = f30_1 - mean(f30_1);
50 u30_2 = f30_2 - mean(f30_2);
51 u30_3 = f30_3 - mean(f30_3);
52 u35_1 = f35_1 - mean(f35_1);
53 u35_2 = f35_2 - mean(f35_2);
54 u35_3 = f35_3 - mean(f35_3);
55
56 U20_1 = u20_1(1:minlength);
57 U20_2 = u20_2(1:minlength);
58 U20_3 = u20_3(1:minlength);
59 U25_1 = u25_1(1:minlength);
60 U25_2 = u25_2(1:minlength);
61 U25_3 = u25_3(1:minlength);
62 U30_1 = u30_1(1:minlength);
63 U30_2 = u30_2(1:minlength);
64 U30_3 = u30_3(1:minlength);
65 U35_1 = u35_1(1:minlength);
66 U35_2 = u35_2(1:minlength);
67 U35_3 = u35_3(1:minlength);
68
69 Fs = 1/16000;
70 WW = 850;
71 fs = 1600;
72
73 %%FINDING THE PSDS AND PLOTTING
74 figure()

```

```

75 text = 'Comparison of the three data sets taken with';
76 [Psd1, F1] = pwelch(U20_1, WW, [], [], fs, 'onesided');
77 loglog(F1, Psd1, 'k')
78 hold on
79 [Psd2, F2] = pwelch(U20_2, WW, [], [], fs, 'onesided');
80 loglog(F2, Psd2, 'k—')
81 [Psd3, F3] = pwelch(U20_3, WW, [], [], fs, 'onesided');
82 loglog(F3, Psd3, 'k:')
83 title([text, 'pump frequency 20 Hz']);
84 legend('first run', 'second run', 'third run');
85
86 figure()
87 [Psd1, F1] = pwelch(U25_1, WW, [], [], fs, 'onesided');
88 loglog(F1, Psd1, 'k')
89 hold on
90 [Psd2, F2] = pwelch(U25_2, WW, [], [], fs, 'onesided');
91 loglog(F2, Psd2, 'k—')
92 [Psd3, F3] = pwelch(U25_3, WW, [], [], fs, 'onesided');
93 loglog(F3, Psd3, 'k:')
94 title([text, 'pump frequency 25 Hz']);
95 legend('first run', 'second run', 'third run');
96
97 figure()
98 [Psd1, F1] = pwelch(U30_1, WW, [], [], fs, 'onesided');
99 loglog(F1, Psd1, 'k')
100 hold on
101 [Psd2, F2] = pwelch(U30_2, WW, [], [], fs, 'onesided');
102 loglog(F2, Psd2, 'k—')
103 [Psd3, F3] = pwelch(U30_3, WW, [], [], fs, 'onesided');
104 loglog(F3, Psd3, 'k:')
105 title([text, 'pump frequency 30 Hz']);
106 legend('first run', 'second run', 'third run');
107
108 figure()
109 [Psd1, F1] = pwelch(U35_1, WW, [], [], fs, 'onesided');
110 loglog(F1, Psd1, 'k')
111 hold on
112 [Psd2, F2] = pwelch(U35_2, WW, [], [], fs, 'onesided');
113 loglog(F2, Psd2, 'k—')
114 [Psd3, F3] = pwelch(U35_3, WW, [], [], fs, 'onesided');
115 loglog(F3, Psd3, 'k:')
116 title([text, 'pump frequency 35 Hz']);
117 legend('first run', 'second run', 'third run');

```

---





# Bibliography

- [1] for Disease Control, C. & Prevention. Measuring blood pressure. URL <http://www.cdc.gov/bloodpressure/measure.htm>. 2015.
- [2] Gupta, R. *et al.* Spectral analysis of arterial sounds: a noninvasive method of studying arterial disease. *Medical and Biological Engineering* (1975).
- [3] Lees, R. S. & Jr., C. F. D. Phonoangiography: A new noninvasive diagnostic method for studying arterial disease. *Proceedings of the National Academy of Sciences* (1970).
- [4] Kim, B. M. & Corcoran, W. H. Experimental measurements of turbulence spectra distal to stenoses. *Journal of Biomechanics* (1974).
- [5] Sabbah, H. N. & Stein, P. D. Turbulent blood flow in humans. *Journal of the American Heart Association* (1976).
- [6] Ferguson, G. G. Turbulence in human intracranial saccular aneurysms. *Journal of Neurosurgery* (1970).
- [7] Merrill, E. W. Rheology of blood. *Physiological Reviews* (1969).
- [8] Reed, T. & Pilehvari, A. A new model for laminar, transitional, and turbulent flow. *Society of Petroleum Engineers, Inc.* (1993).
- [9] Pope, S. B. *Turbulent Flows* (Cambridge University Press, 2000), 8th edn.
- [10] Camussi, R. (ed.) *Noise Sources in Turbulent Shear Flows: Fundamentals and Applications* (Springer, 2013).
- [11] Hayes, M. H. *Statistical Digital Signal Processing and Modeling* (John Wiley & sons, Inc., 1996).

- [12] MathWorld. Power spectrum. URL <http://mathworld.wolfram.com/PowerSpectrum.html>. 2015.
- [13] George, W. K. Lectures in turbulence for the 21st century. <http://www.turbulence-online.com> (2013).
- [14] Ambardar, A. *Digital Signal Processing: A Modern Introduction* (Thomson, 2007). URL <https://books.google.no/books?id=SJqvPQAACAAJ>.
- [15] Mathworks. Documentation on fft. URL <http://se.mathworks.com/help/matlab/ref/fft.html>. 2015.
- [16] Kay, S. M. *Modern Spectral Estimation: Theory and application* (Prentice Hall, 1988).
- [17] Instruments, N. "what is ni mydaq" (software). URL <http://www.ni.com/mydaq/what-is/>. 2015.
- [18] Instruments, N. Labview (software). URL <http://www.ni.com/labview/>. 2015.