Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

Henrik K. R. Berg

In-situ analysis of room modes and absorption coefficients at low frequencies

Master's thesis in Electronic Systems Design
Supervisor: Peter Svensson
Co-supervisor: Jens Holger Rindel
June 2021

NTNU
Norwegian University of Science and Technology

Henrik K. R. Berg

In-situ analysis of room modes and absorption coefficients at low frequencies

Master's thesis in Electronic Systems Design
Supervisor: Peter Svensson
Co-supervisor: Jens Holger Rindel
June 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

NTNU
Norwegian University of
Science and Technology

In-situ analysis of room modes and absorption coefficients at low frequencies

Henrik K. R. Berg

2021/06/15

Preface

This thesis was first and foremost made possible through cooperation with Multiconsult. I want to thank Ingunn Milford, the head of the acoustic department at Multiconsult, that made the project possible and made their facilities and measurement equipment available. I also want to thank my supervisors, Peter Svensson at the Norwegian University of Science and Technology, and Jens Holger Rindel at Multiconsult. It has been a privilege to be supervised by such knowledgeable and experienced acousticians. It has also been a pleasure testing out the prototype loudspeaker Jens provided through Odeon.

Furthermore, I want to thank Erik Arvidsson at Multiconsult. Erik contributed a lot by assisting me during the measurements and allowing me access to their facilities when I needed additional measurements. Additionally, I want to thank Huy Pham at Multiconsult for providing information about the rooms' construction and Claus Lynge Christiansen at Odeon for showing how to retrieve the attenuation factor hidden within the wave files exported by Odeon.

Abstract

In-situ measurements of a wall's acoustic properties at low frequencies are a relatively unexplored topic. In this thesis, two measurement methods have been explored and tested. Microphone and loudspeaker positions are used to approach plane wave propagation at low frequencies. Then the absorption coefficient is estimated from the standing wave ratio between two opposing walls or through the modal reverberation time. A least-squares-fit model has also been established to counter the spatial resolution of the standing wave patterns measured. Unfortunately, the absorption coefficient obtained through the least-squares-fit model was very different from the ones calculated through the two measurements mentioned. Although the estimated absorption coefficients from the different measurement methods are of similar magnitudes, there is no way of knowing whether the measurement values are correct. In order to determine the validity of the measurements, further research is needed.

Sammendrag

In-situ målinger av en veggs akustiske egenskaper ved lave frekvenser er et relativt uutforsket emne. I denne oppgaven har to målemetoder blitt utforsket og testet. Posisjonene til mikrofon og høyttaler er brukt til å tilnærme seg planbølgeforplantning ved lave frekvenser langs én akse i rommet. Absorpsjonskoeffisienten er da både beregnet fra det stående bølgeforholdet, og fra den modale etterklangstiden. En minste kvadraters metode har også blitt brukt i forsøk på å motvirke den lave romlige oppløsningen til de målte stående bølgene. Absorpsjonskoeffisienten beregnet via minste kvadraters metoden har uheldigvis hatt veldig forskjellige verdier fra de andre to nevnte målemetodene. Selv om verdiene fra de to førstnevnte metodene i seg selv har verdier av liknende størrelsesorden, så har man ikke kunnet si noe om hvor korrekte verdiene er. For å kunne vurdere dette, må målemetodene utforskes videre.

Contents

Chapter 1

Introduction

1.1 Motivation

In small music rehearsal rooms, having a smooth frequency response is essential so that different tones are evenly supported by the room [1]. The density of natural frequencies is lower in small rooms in the lower frequencies, and it becomes vital that the natural room modes are evenly spaced out to have a smooth frequency response. The shape of normal modes and corresponding natural frequencies are well known for shoebox-shaped rooms with rigid walls. However, when one or more walls have a complex impedance deviating substantially from a rigid wall, these parameters change and become harder to predict. A complex

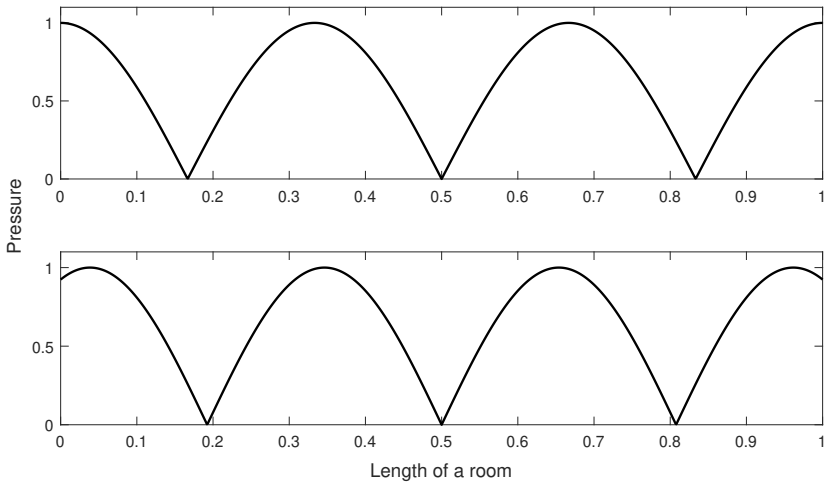


Figure 1.1: Example of how a purely imaginary impedance can shift the phase of the reflected wave.

wall impedance may not only introduce absorption at the wall, but it can introduce a phase shift of the reflection, moving the pressure maximum from the wall

to somewhere in front of the wall. Figure 1.1 illustrates how a purely complex impedance would cause a phase shift and move the pressure maximum into the room. The wavelength of that particular room resonance has thus become shorter. The impedance of these walls at lower frequencies would need to be estimated to predict the room's behavior better.

1.2 Background

In 1992, R. Walker tried to predict the frequency response in a room but concluded that the frequency error in the room's resonances became too significant at a given frequency due to the unknown wall impedance. In 1944, P. M. Morse and R. H. Bolt [2] stated that as the impedance varies with frequency, the boundary conditions will be different for each standing wave in the room, and thus not form an orthogonal set of characteristic functions. Thus the usual methods of predicting the room response cannot be used. They also stated that the wavenumber and attenuation parameters have to be solved experimentally for impedance boundary conditions. Today there are efficient ways of determining these parameters numerically [3], and information about the impedance of the wall can easier be used to estimate natural frequencies in the room. A company called Microflown has developed an in-situ technique and a probe to measure the acoustic absorption [4], reflection, or impedance of a material. These properties can be determined at both normal and oblique angles, but unfortunately, the method is only usable between 300 Hz and 10 kHz.

1.3 Problem description

The Microflown method cannot be used at lower frequencies, and the usual methods of measuring the impedance of a surface are carried out in either an impedance tube, [5][6], or in a reverberation chamber [7], which becomes impractical at low frequencies and are irrelevant for in-situ situations. Thus, other ways of determining the acoustic properties of a room's walls need to be discovered. A. Celestinos and S. B. Nielsen [8] showed that a plane wave could be created in a room by clever positioning of source and receiver. The impedance tube method approaches plane wave propagation by using a tube where the cross-section is small compared to the length of the tube. Thus, using a similar method to Celestinos and Nielsen, an attempt to approach a plane wave has been made. The impedance tube methods might be applicable by approaching a plane wave along one axis in a room. In an attempt to measure the impedance of the walls in an arbitrary shoebox-shaped room, clever loudspeaker and microphone positions have been used to isolate axial modes to approach a plane wave by preventing waves from traveling in any other direction within the same frequency range. This method could then make the room behave like a large impedance tube for the lowest axial modes. Clever source and receiver positions have been used to measure dif-

ferent modal reverberation times and standing wave patterns in the rooms. The modal reverberation time has been used to obtain a modal absorption coefficient for a wall, although it cannot obtain the wall impedance. The measurement of the standing waves has been used to estimate the complex reflection factor at the modal center frequencies, which can be converted to either wall impedance or absorption coefficient. Furthermore, a least-squares-fit model of the measured standing wave is created to see if the results can be enhanced. As the absorption coefficient is the only common parameter, this parameter will be compared between the different methods to discuss how well these experimental measurement techniques work.

1.4 Outline

The theory relevant to this thesis will be laid out in Chapter 2, which includes how the absorption coefficient can be calculated from the two different measurement methods, how to calculate a least-squares-fit model of the standing wave and how to estimate the wall's impedance at low frequencies. In Chapter 3, the measurement setup and the measurement procedure is described. Also, the rooms are thoroughly described, and some of the post-processing is explained. The results of the measurements through post-processing and calculations made from the collected data are then examined in Chapter 4. The different frequency responses, the standing wave patterns measured and modeled, the decay curves and their parameters, and the estimated absorption coefficients are shown in this chapter. The results obtained by different methods are then compared and discussed in Chapter 5 before the conclusive remarks are drawn in Chapter 6.

Chapter 2

Theory

In this chapter, the theory behind both the measurement methods and calculations performed are presented. Firstly the room acoustic parameters that were attempted measured and their relationship is presented. These parameters involve the surface impedance, reflection factor, and absorption coefficient. A simplified method of estimating the wall properties based on the wall construction is presented too. Then, the methods used to estimate the wall parameters off the measured impulse responses rely heavily on plane wave propagation. How plane wave propagation within a limited frequency range has been approached, and the method for doing so is also explained in this chapter and how the plane wave's standing wave pattern can determine a wall's properties. A numerical method for determining the least-squares-fit coefficients for a one-dimensional plane wave equation to fit the measured pattern is also presented. The idea is that the least-squares-fit model might compensate for the spatial resolution of the measured standing wave pattern and artifacts caused by energy propagating in different directions than normal to the opposing surfaces. The second method presented is trying to measure the absorption coefficient of the surfaces through its relationship with modal reverberation time. The method of obtaining the modal reverberation time and its use to calculate the absorption coefficient is explained in this chapter.

2.1 Impedance, reflection and absorption

The surface impedance $\mathbf{Z}$ is a complex ratio between the sound pressure $\mathbf{p}$ and the particle velocity normal to a surface v_n [2].

$$\mathbf{Z} = \left(\frac{\mathbf{p}}{v_n}\right)_{surface} \tag{2.1}$$

The particle velocity is caused by vibrations in the wall or by air moving into pores in the wall. If the wall impedance has a real part, energy will be absorbed. If the wall impedance is purely imaginary, it only causes a phase shift in the wave's reflection. For more convenient expressions, the normalized specific acoustic im-

pedance on the wall will be used, which is the wall impedance divided by the characteristic impedance of air [9], $\zeta = Z/\rho_0 c$.

If an incident plane sound wave hits a wall with a normalized specific acoustic impedance ζ, the ratio between the amplitude of the incoming sound wave and the reflected one is given as:

$$\mathbf{r} = \frac{\zeta cos\theta - 1}{\zeta cos\theta + 1} \tag{2.2}$$

where θ is the incident angle of the sound wave. The amount of energy lost upon reflection is given by the absorption coefficient:

$$\alpha = 1 - |\mathbf{r}|^2 \tag{2.3}$$

2.2 Impedance of a finite plate

As there is little to no information available about impedance measurement at low frequencies, it becomes harder to evaluate the results of the measurements. Thus, estimation of the surfaces' impedance and absorption coefficient could help evaluate the measured values.

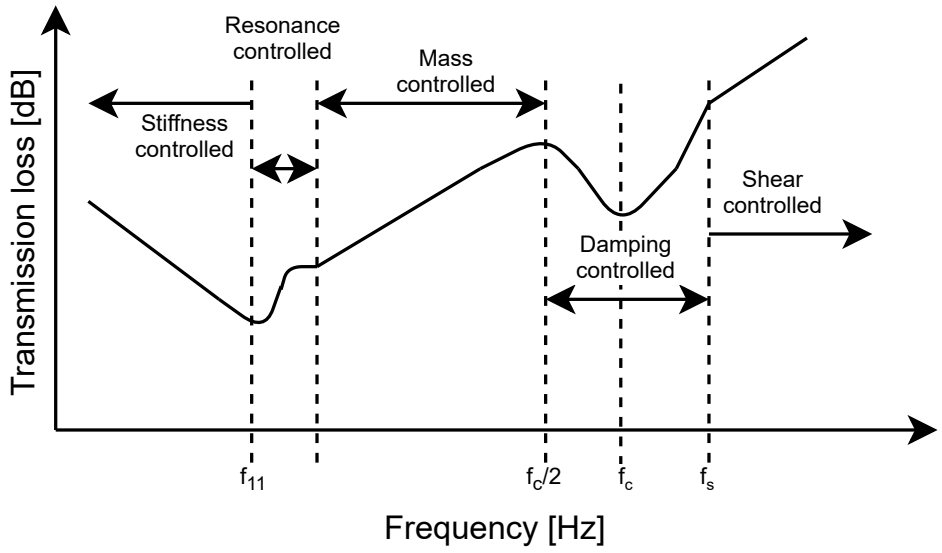


Figure 2.1: Transmission loss versus frequency for a thin panel. Figure inspired by Figure 9.15 in [10].

At lower frequencies, below the critical frequency, a thin panel acts as one moving mass [10]. The critical frequency is given as:

$$f_c = \frac{c_0^2}{2\pi h}\sqrt{\frac{12(1-\sigma^2)\rho_m}{E}} \tag{2.4}$$

where

c_0 is the speed of sound in air, in meters per second;

h is the thickness of the material, in meters;

σ is the Poisson's ratio of the material;

ρ_m is the mass density of the material, in kilograms per cubic meter;

E is the material's Young's modulus, in Pascal.

The mass density, Poisson's ratio, and Young's modulus for glass and gypsum can be found in Table 2.1.

At an even lower frequency, the natural frequency of the first fundamental structural mode is found. Around and below this frequency, the impedance is dominated by the plate's stiffness, and boundary connections [11]. This fundamental frequency is given by equation Equation 2.5, which are valid for a plate surrounded by an infinite baffle [12].

$$f_{11} = \frac{c_0^2}{4f_c}\left(\left(\frac{1}{l_x}\right)^2 + \left(\frac{1}{l_y}\right)^2\right) \tag{2.5}$$

where l_x and l_y is the plate dimensions, excluding its thickness.

As only sound waves with normal incidence on the surfaces will be considered, the wall impedance below the critical frequency is given as by Equation 2.6, where the real part works as damping in a mass-spring system.

$$Z_w = j\omega m\left(1 - \left(\frac{f_{11}}{f}\right)^2\right) + \eta\omega m \tag{2.6}$$

where

$\omega = 2\pi f$ is the angular frequency, in radians;

$m = \rho_m h$ is the surface mass density, in kilograms per square meter;

$\eta = \eta_{tot}$ is the loss factor.

The loss factor describes the energy losses in vibrations and is dependent on quite a few parameters. The contribution to the total loss factor η_{tot}, comes from internal losses η_{int}, boundary losses η_{border}, and radiation losses η_{rad}, as described in [11]. The internal losses, which are nearly frequency independent, are caused by sound energy converted to heat energy. The internal losses for glass and gypsum are listed in Table 2.1. The radiation losses are caused by sound radiated from the plate, and the boundary losses are sound energy transferred to connected structures. The total loss factor is then given as:

$$\eta_{tot} = \eta_{int} + \eta_{border} + \eta_{rad} = \eta_{int} + \frac{Uc_0\alpha}{\pi^2 S}(f \cdot f_c)^{-1/2} + \frac{2\rho_0 c_0}{\omega m}\sigma_{res} \tag{2.7}$$

where

U is the perimeter of the plate, in meters;

S is the surface area of the plate, in square meters;

α_s is the average structural absorption coefficient along the boundary;

σ_{rad} is the resonant radiation efficiency.

Table 2.1: Properties of glass and gypsum. Data taken from Table 3.1 in [13]. Poisson's ratio of glass taken from [14].

Material	**Density** kg/m^3	**E-modulus** 10^9Pa	**Poisson's ratio**	**Loss factor** $\eta_{int} \cdot 10^{-3}$
Glass	2500	60	∼0.2	0.6-2.0
Gypsum plate	800-900	4.1	∼0.3	10-15

2.3 Natural frequencies in cuboids with rigid surfaces.

Although the low-frequency behavior of non-rigid walls is of interest, the easier and well-known solutions for a room with rigid surfaces are helpful. These can be used to find the approximate location of the natural frequencies and reveal which modes are adjacent in the frequency domain.

Helmholtz equation as seen in Equation 2.8, is a time-independent form of the wave equation which relates sound pressure in space. A time-harmonic factor $e^{j\omega t}$ has been assumed.

$$\nabla^2 \mathbf{p} + k^2 \mathbf{p} = 0 \tag{2.8}$$

where ∇^2 is the laplace operator, $k = \frac{\omega}{c}$ is the wavenumber and p is the sound pressure. The general solution to the Helmholtz equation in a cuboid cavity with rigid boundaries is a summation of characteristic equations:

$$\mathbf{p}(x,y,z) = \sum_{n_x} \sum_{n_y} \sum_{n_z} \mathbf{p}_{n_x n_y n_z}(x,y,z) \tag{2.9}$$

Each characteristic equation is then given as:

$$\mathbf{p}_{n_x n_y n_z}(x,y,z) = \mathbf{A}_{n_x n_y n_z} \psi_{n_x n_y n_z} = \mathbf{A}_{n_x n_y n_z} cos(k_x x) cos(k_y y) cos(k_z z) \tag{2.10}$$

where $\mathbf{A}_{lmn}$ is the comlex amplitude of the mode function, x, y and z are cartesian coordinates and the constants k_x, k_y and k_z are given by Equation 2.11. The time harmonic factor $e^{j\omega t}$ has been left out of Equation 2.10 for simplicity.

$$\begin{aligned} k_x &= \frac{n_x \pi}{L_x} \quad n_x = 0,1,2,3,... \\ k_y &= \frac{n_y \pi}{L_y} \quad n_y = 0,1,2,3,... \\ k_z &= \frac{n_z \pi}{L_z} \quad n_z = 0,1,2,3,... \end{aligned} \tag{2.11}$$

where n_z, n_y and n_z is number of nodal planes perpendicular to each axis, and L_x, L_y and L_z is the length of each dimension of the cuboid.

The expected modal frequencies in a rigid cuboid, which is the center frequency of the respective mode function, is then given by the following equation:

$$f_n = \frac{c_0}{2}\sqrt{\left(\frac{n_x}{L_x}\right)^2 + \left(\frac{n_y}{L_y}\right)^2 + \left(\frac{n_z}{L_z}\right)^2} \tag{2.12}$$

It is also possible to calculate the transfer function of the room between two points, e.g. a source and receiver. This is quite useful in combination with Equation 2.12 such that the measured frequency response in a room can be compared to calculations, which in turn can help identifying the different resonance frequencies in a room. The transfer function is taken from [15] and looks as follows:

$$\mathbf{p}_\omega(x,y,z) = \frac{j\mathbf{U}(\omega)c_0^2\omega\rho_0}{V}\sum_n \frac{1}{\epsilon_n}\frac{\psi_n(x,y,z)\psi_n(x_0,y_0,z_0)}{\omega^2-\omega_n^2-2j\delta_n\omega_n} \tag{2.13}$$

where

n	is unique combinations of n_x, n_y and n_z;
ψ_n	is the mode shape function, found through Equation 2.10;
(x,y,z)	is the coordinates of the source;
(x_0,y_0,z_0)	is the coordinates of the receiver;
$\mathbf{U}(\omega)$	is the volume velocity of the sound source;
$\delta_n = \frac{2.2\pi}{T_n}$	is the modal damping based on the modal reverberation time;
ω_n	is the angular natural frequency of mode n;
ϵ_n	is a mode normalization factor, which is equal to $1/2$ for axial modes, $1/4$ for tangential modes and $1/8$ for oblique modes.

2.4 Standing wave pattern

At the frequencies given by Equation 2.12, a standing wave pattern is formed inside the room. Figure 2.2 displays how the absolute sound pressure for the first five axial modes between two rigid surfaces. An axial mode will only have a non-zero values for one of n_x, n_y, and n_z. The nodes are then the points of maximum destructive interference, which in the case of two opposing rigid walls, the sound pressure would be zero. These occur when the respective cosine term in Equation 2.10 goes to zero. If a point source is positioned in one of these nodes, the source can not excite the respective modes. The same goes for a point receiver. If it is placed exactly in a node, it will not receive any signal for the given frequency. In a three-dimensional case, these nodal points become nodal planes normal to the respective axis.

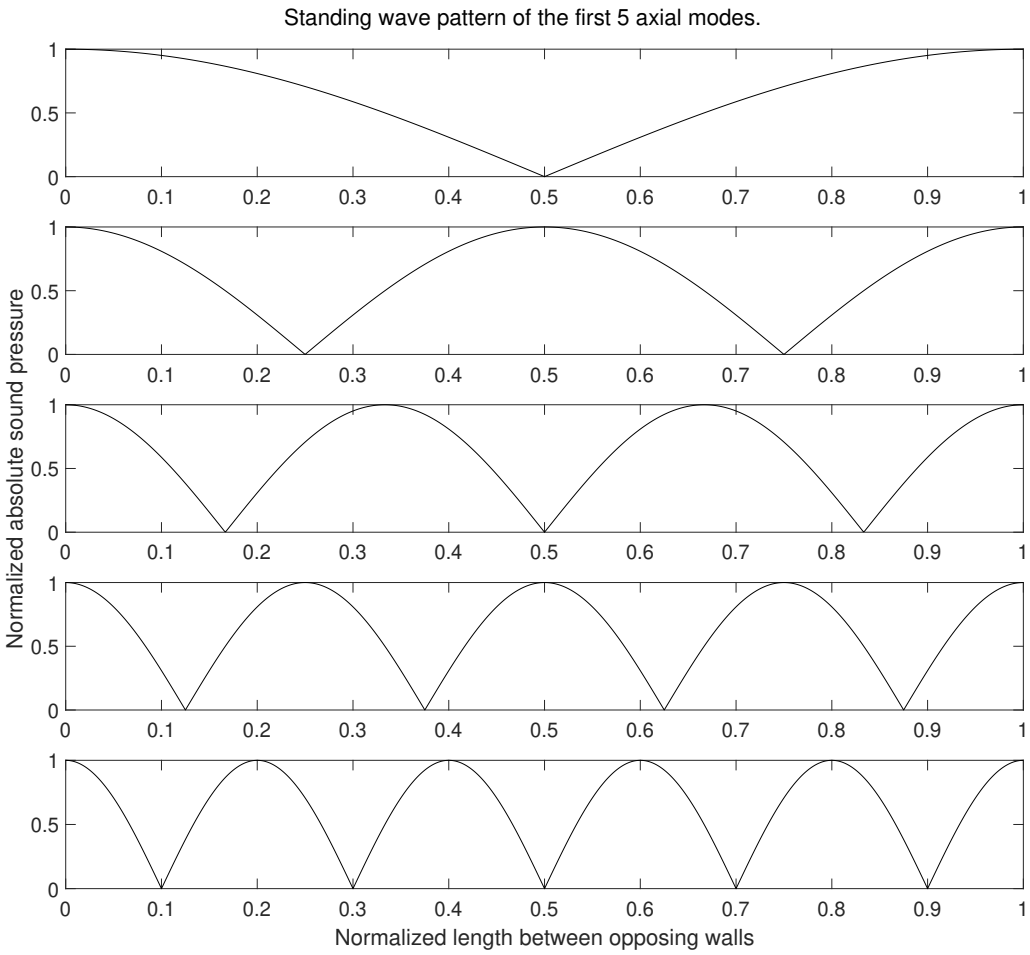


Figure 2.2: Standing wave pattern of the first five axial modes between two rigid surfaces.

2.5 Plane wave approach

The sound propagation needs to be as close to plane wave propagation as possible to justify the methods used to estimate the wall's properties. As described in [8], a plane wave can be simulated by only exiting the axial modes in one direction, within a limited frequency range. Although it is optimistic making the other modes cease to exist, it is possible to reduce several modes' amplitude severely. This reduction can be achieved by positioning the source and receiver along an intersection between two perpendicular nodal planes, suppressing any mode with a pressure node on this intersection.

An example of such source and receiver positions can be seen in Figure 2.3. A

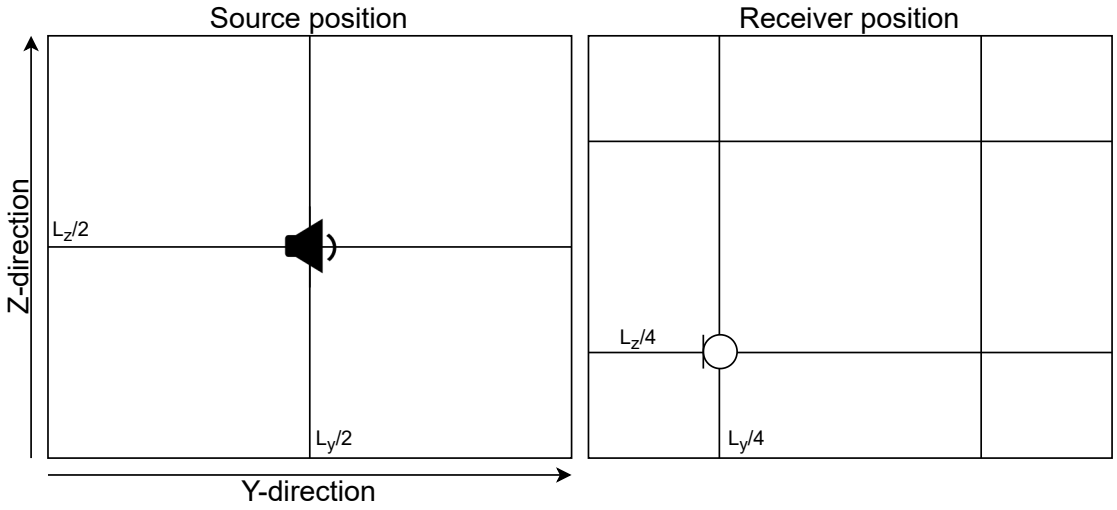


Figure 2.3: Position of source and receiver at intersection between nodal lines for a (n_x,1,1) mode (left), and a (n_x,2,2) mode (right).

loudspeaker is placed in $(L_y/2, L_z/2)$, preventing any mode containing n_y or n_z equal an odd number from being excited. It is also possible to see in Figure 2.2 that the first, third, and fifth modes all have nodes at the halfway point between two walls. A receiver positioned in $(L_y/4, L_z/4)$ would prevent any mode with n_y or n_z equal to two. In a rigid cuboid room with a point source and point receiver, a configuration like in Figure 2.3 would then isolate the first few axial modes in the x-direction.

It is still important to realize that the wavelength and locations of pressure nodes may change if the wall has a complex impedance. The node located in the room's center for odd modes may also move if opposing walls have unequal impedance. Thus, in a room with unknown impedance at the walls, these nodal points must be discovered experimentally, although they are expected to be somewhere near the theoretical location in an equivalent rigid room.

2.6 Standing wave ratio

The method of obtaining the reflection factor, absorption coefficient, and surface impedance from the standing wave ratio is adapted from the standing wave ratio method for impedance tube measurements [5]. This method relies on only a plane incident wave and a plane reflected wave. Thus, in this case, it will only be used where the measured standing wave ratio takes the appropriate shape, which would be close to Figure 2.2. The standing wave ratio, which is the ratio between the maximum and the minimum pressure of a standing wave, is given as:

$$s = \frac{|p_{max}|}{|p_{min}|} \tag{2.14}$$

where

$|p_{max}|$ is the first pressure maximum, moving away from the surface;

$|p_{min}|$ is the first pressure minimum, moving away from the surface.

The modulus of the reflection factor is then given as:

$$|R_p| = \frac{s-1}{s+1} \tag{2.15}$$

The phase angle of the reflection factor is then given as:

$$\phi = \pi(\frac{4x_{min,1}}{\lambda_1} - 1) \tag{2.16}$$

where $x_{min,1}$ is the coordinate of the first pressure minimum when moving away from the surface of interest.

The wavelength of that particular frequency is then either found by the relation $\lambda_0 = c_0/f$ or by the distance between two neighboring pressure minima:

$$\lambda_0 = 2(x_{min,2} - x_{min,1}) \tag{2.17}$$

After converting the reflection factor from a polar to a complex value, it can then be used with both Equation 2.3 to find the absorption coefficient and with Equation 2.2 to find the normalized specific impedance of the wall at normal incidence.

2.7 Modelling the standing wave

Although it might be enough to input the measurement data directly into the Equation 2.14-2.17, the measured data is prone to the spacing between each microphone position and sound energy propagating in non-axial directions. The microphone positions used will not necessarily be positioned in a pressure minimums exact location, and the standing wave ratio calculated from the measured values might be inaccurate. A least-squares-fit plane wave will be modeled based on the measurement data to attempt to be less susceptible to such errors.

The general solution to the wave equation in one dimension is given as:

$$p(x,t) = (\mathbf{A}e^{-jkx} + \mathbf{B}e^{jkx})e^{j\omega t} \tag{2.18}$$

where **A** and **B** are the complex amplitudes of two plane waves traveling in opposite directions along the x-axis. This expression can be used to model the measured standing wave pattern, that is, if the measured data reassembles a standing wave pattern in the first place. Due to the measurement equipment used, the initial time delay of the measured signal's direct sound is lost, and the only valuable measured data is $|p_{measured}|$. The expression needs to be altered as the amplitude **B** can

not be recovered. The square pressure can then be modeled using the following expression:

$$|p(x)_{model}|^2 = |A|^2(1 + |R_p|^2 + 2R_p cos[2kx + \phi]) \tag{2.19}$$

where

$\mathbf{R}_p = \mathbf{B}/\mathbf{A}$ is the reflection factor;

$\phi = arg(R_p)$ is the phase angle of the reflection factor;

k is the wavenumber.

The best-fitting values to Equation 2.19 for a given frequency is then found using a fminsearch algorithm in Matlab, which is based on the Nelder-Mead simplex algorithm [16], which in this case solves for the lowest possible value of ϵ^2:

$$\epsilon^2 = \sum \left(|p_{model}|^2 - |p_{measurement}|^2\right) \tag{2.20}$$

The algorithm needs an initial guess of each value as its starting point. These initial guesses will be based off Equations 2.15, 2.16 and 2.19, and give the following three equations:

$$|A|^2_{initial\ guess} = \frac{|p(x)_{max,measurement}|^2}{(1 + |R_p|_{initial\ guess})^2} \tag{2.21}$$

$$|R_p|_{initial\ guess} = \frac{s-1}{s+1} \tag{2.22}$$

$$\phi_{initial\ guess} = 2kx_{min,1} - \pi \tag{2.23}$$

The fminsearch algorithm then finds the values of $|A|^2, |R_p|$ and ϕ that returns the lowest values for ϵ^2. In other words, a least-mean-squares solution to Equation 2.19, which models the measured standing wave pattern. The process then needs to be repeated for all frequencies of interest.

2.8 Absorption coefficient from modal reverberation time

In order to measure the modal reverberation times in a room, the mode of interest needs to be isolated as much as possible so that no other mode influences the reverberation time obtained. It can be isolated by combining the source and receiver's clever positioning and fitting bandpass filters.

The source and receiver positioning will be by the same principles as in section 2.5, as it is the axial modes that are of interest. However, in this case, the goal is to isolate a single mode, and the source and receiver must account for all three dimensions when placed to reduce the influence of even more modes. Although usually, rooms do not have rigid walls, Equation 2.12 can help determine which room modes which will be in the vicinity to each other in the frequency domain.

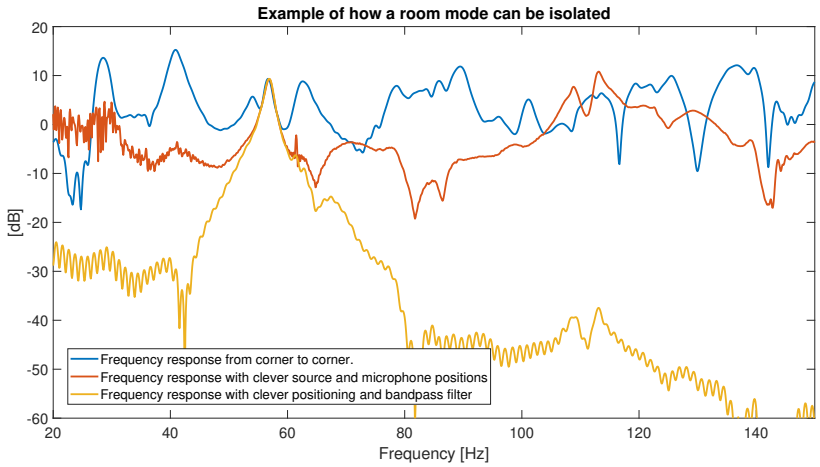


Figure 2.4: Example of how a room mode can be isolated through source and receiver positioning as well as bandpass filtering.

Table 2.2: Expected modal frequencies in the rooms studied.

Room A				Room B				Room C			
n_x	n_y	n_z	f_n	n_x	n_y	n_z	f_n	n_x	n_y	n_z	f_n
1	0	0	27.2 Hz	1	0	0	49.0 Hz	0	0	1	50.4 Hz
0	1	0	27.9 Hz	0	0	1	50.4 Hz	1	0	0	58.5 Hz
1	1	0	39.0 Hz	1	0	1	70.3 Hz	0	1	0	74.6 Hz
0	0	1	50.4 Hz	0	1	0	78.3 Hz	1	0	1	77.3 Hz
2	0	0	54.4 Hz	1	1	0	92.4 Hz	0	1	1	90.0 Hz
0	2	0	55.8 Hz	0	1	1	93.1 Hz	1	1	0	94.8 Hz
1	0	1	57.3 Hz	2	0	0	98.0 Hz	0	0	2	100.9 Hz
0	1	1	57.6 Hz	0	0	2	100.9 Hz	1	1	1	107.4 Hz
2	1	0	61.2 Hz	1	1	1	105.3 Hz	1	0	2	116.6 Hz
1	2	0	62.1 Hz	2	0	1	110.2 Hz	2	0	0	117.1 Hz
1	1	1	63.7 Hz	1	0	2	112.2 Hz	0	1	2	125.4 Hz
2	0	1	74.2 Hz	2	1	0	125.4 Hz	2	0	1	127.5 Hz
0	2	1	75.2 Hz	0	1	2	127.7 Hz	1	1	2	138.4 Hz
2	2	0	77.9 Hz	2	1	1	135.2 Hz	2	1	0	138.8 Hz
2	1	1	79.3 Hz	1	1	2	136.8 Hz	2	1	1	147.7 Hz
1	2	1	80.0 Hz	2	0	2	140.6 Hz	0	2	0	149.1 Hz
3	0	0	81.7 Hz	3	0	0	147.0 Hz	0	0	3	151.3 Hz
0	3	0	83.7 Hz	0	0	3	151.3 Hz	2	0	2	154.5 Hz
3	1	0	86.3 Hz	3	0	1	155.4 Hz	0	2	1	157.4 Hz
1	3	0	88.0 Hz	0	2	0	156.6 Hz	1	2	0	160.2 Hz
2	2	1	92.8 Hz	1	0	3	159.1 Hz	1	0	3	162.2 Hz
3	0	1	96.0 Hz	2	1	2	161.0 Hz	1	2	1	168.0 Hz
0	3	1	97.7 Hz	1	2	0	164.1 Hz	0	1	3	168.7 Hz
3	2	0	98.9 Hz	0	2	1	164.6 Hz	2	1	2	171.6 Hz
2	3	0	99.8 Hz	3	1	0	166.6 Hz	3	0	0	175.6 Hz

When trying to isolate a mode, i.e., mode $(n_x, n_y, n_z) = (2,0,0)$, the nearby modes in the frequency domain need to be canceled to the best extent. The modal shape function for a given source or receiver position in a rigid room is given by Equation 2.10. The source or receiver needs to be positioned in one of the room's pressure nodes, such that one of the cosines becomes 0, which suppresses the mode. For instance, to cancel a (1,0,0) mode, either the source or receiver needs to be positioned along $x = l_x/2$. In other words, a source placed in the exact center of the room will cancel all modes containing any odd mode number. However,

in a more realistic room where the walls have a complex impedance, the nodes may have shifted slightly. Thus a few measurements are needed to find the actual nodal lines in the room. Then using the information gathered from Table 2.2, Equation 2.10 and the position of the modal lines found experimentally, several nearby modes might be canceled. The used source and receiver positioning will be presented in the next chapter.

After the influence of the nearby modes is reduced, a bandpass filter can be applied to reduce the influence from room modes further away in terms of frequency. The bandwidth of the filter should not be too narrow, as the reverberation time of the filter itself can influence the results, according to [17]. The article also states that using a reverse filtering technique can reduce the filter's distortion imposed on the signal. Reverse filtering can be achieved as easily as flipping the impulse response backward before convolving it with the filter, then flipping it back afterward. A bandpass filter's bandwidth is then acceptable if it fulfills the equality:

$$B \cdot T_{60} > 4 \tag{2.24}$$

where B is the filter's bandwidth in hertz and T_{60} is the reverberation time. The effect the transducer positioning and the bandpass filter can have on the signal is illustrated in Figure 2.4. The impulse response should then ideally only oscillate with one frequency as seen in Figure 2.5.

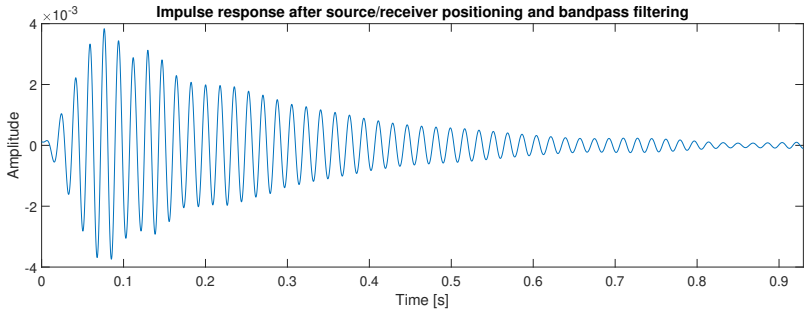


Figure 2.5: Example of how the impulse response may look after one mode is isolated.

After the bandpass filter has been applied, a mode of interest would ideally be the only thing left in the impulse response. The next step to obtain the modal reverberation time is to calculate the energy decay curve. The method used to obtain a decay curve is taken from [18] and is based on backward integration of a truncated squared impulse response.

As shown in Figure 2.6, the impulse response vanishes into background noise at some point, and this background noise is unwanted during the backward integration. It is suggested to start the backward integration from the point where the impulse response is 10 dB above the noise floor [18]. In Figure 2.6, that would be where the dashed blue line is 10 dB above the dashed pink line. The energy decay curve is then obtained by backward integration according to Equation 2.25.

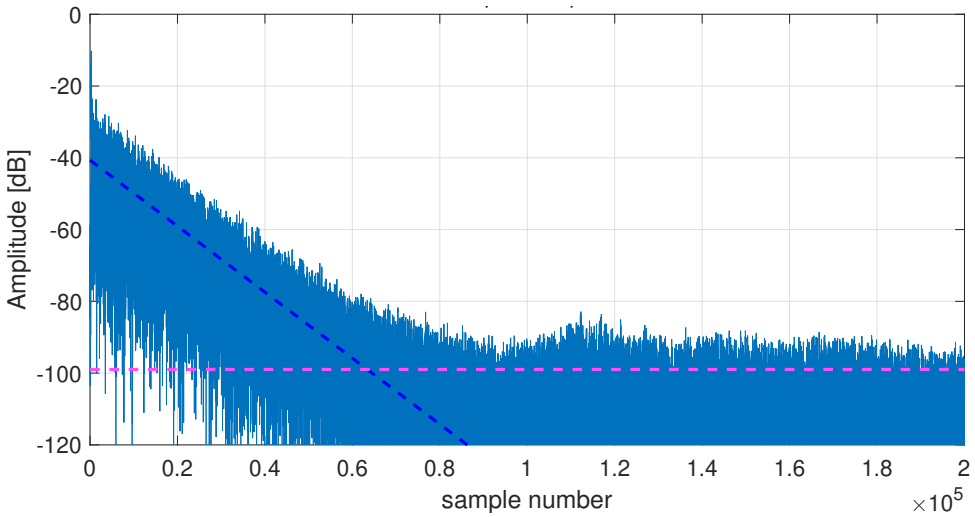


Figure 2.6: A filtered impulse response where the slope of the squared impulse response is illustrated by the dashed blue line, and the mean background noise level is illustrated by the pink line.

$$E(t) = \int_{t_1}^{t} p^2(\tau) d(-\tau) \tag{2.25}$$

where t_1 is the point in time where the squared impulse response is 10 dB above the noise floor.

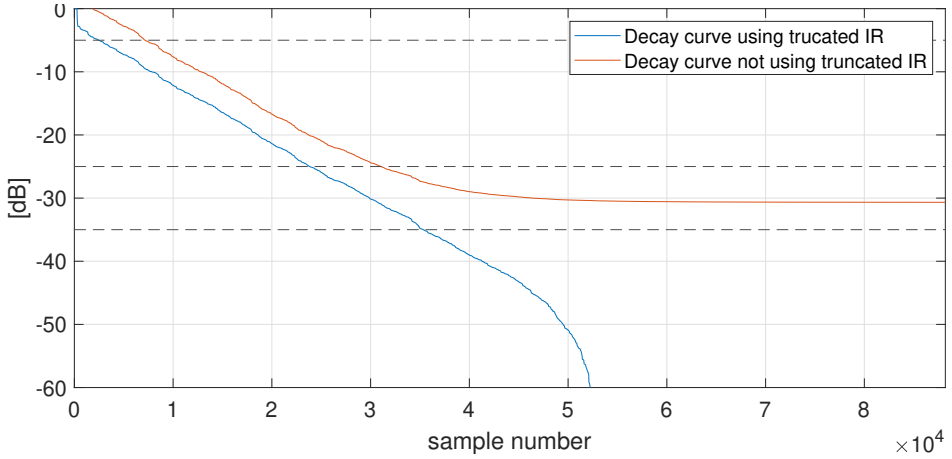


Figure 2.7: Decay curve of a backwards integrated impulse response with and without truncation. The dashed lines marks the levels -5, -25 and -35 dB.

The energy decay curve obtained from the impulse response in Figure 2.6 can be seen in Figure 2.7 on a logarithmic y-axis. The orange line shows what happens if the backward integration starts at the end of the impulse response instead of at

t_1. The noise becomes part of the integration and conceals parts of the decay. On the other hand, as can be seen in Figure 2.7, the decay curve from the truncated impulse response rolls off before it has decayed by 60 dB. Thus linear regression must be applied to compensate for the truncation effect. A straight least-squares fit line is calculated from all points between where the decay curve has dropped 5 dB and 25 or 35 dB. These are the evaluation ranges for the reverberation times T_{20} and T_{30}, and are illustrated in Figure 2.8. The reverberation time is then the time it takes the least-squares fit line to decrease by 60 dB.

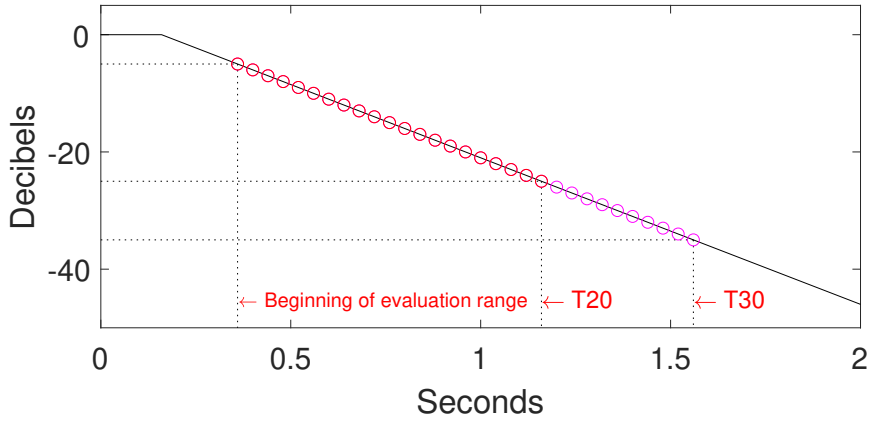


Figure 2.8: Decay curve with evaluation range for T20 and T30

As a single normal mode in the room should have an exponential decay rate, it is expected that the decay curve with a logarithmic y-axis is straight. Decay curves are not necessarily as linear as in Figure 2.7, and the linearity of the decay curve needs to be established. The degree of non-linearity, ξ, describes the decay curve's deviation from the least-squares fit line in per mille [19]. A deviation of more than 10‰ indicates that the decay curve is far from straight within the evaluation range. The degree of non-linearity is given as:

$$\xi = 1000(1 - r^2) \tag{2.26}$$

where r is the correlation coefficient, which is given as:

$$r^2 = \frac{\sum_{i=1}^{n}(\hat{L}_i - \bar{L})^2}{\sum_{i=1}^{n}(L_i - \bar{L})^2} \tag{2.27}$$

where

L_i is the level of the decay curve, in decibels;

$\bar{L}$ is the mean level of all samples within the evaluation range, in decibels;

$\hat{L}_i$ is the level of the linear regression, in decibels;

i is the sample number within the evaluation range;

After obtaining the modal reverberation time, the absorption coefficient of the surfaces involved can be determined. The relationship between the modal reverberation time T_n and absorption coefficients of each surface [20] is given by Equation 2.28. The expression is based on the fact that a standing wave of an oblique mode in a cuboid room can be split into eight plane waves.

$$T_n = \frac{55.3 \cdot V \cdot f_n}{-c^2} \cdot \left[\frac{n_x}{l_x} S_x ln((1-\alpha_{x1})(1-\alpha_{x2})) + \frac{n_y}{l_y} S_y ln((1-\alpha_{y1})(1-\alpha_{y2})) + \frac{n_z}{l_z} S_z ln((1-\alpha_{z1})(1-\alpha_{z2})) \right]^{-1} \tag{2.28}$$

where

T_n is the modal reverberation time, in seconds;

$V = l_x l_y l_z$ is the room volume, in cubic meters;

$S_x = l_y l_z$ is the surface area of walls parallel to the x-axis, in square meters;

$S_y = l_x l_z$ is the surface area of walls parallel to the y-axis, in square meters;

$S_z = l_x l_y$ is the surface area of walls parallel to the z-axis, in square meters;

l_x, l_y & l_z is the length of each dimension of the room, in meters;

α_{x1} & α_{x2} is the absorption coefficients of the walls parallell to the x-axis;

α_{y1} & α_{y2} is the absorption coefficients of the walls parallell to the y-axis;

α_{z1} & α_{z2} is the absorption coefficients of the walls parallell to the z-axis;

Chapter 3

Method

A series of impulse response measurements were carried out to obtain information about a room's absorption coefficient or specific acoustic impedance. The first approach is based on suppressing several room modes by positioning the microphone and loudspeaker on the respective nodes. Such positioning, together with some post-processing, as explained in section 2.8, can be used to isolate a single room mode. From such measurement, the modal reverberation time can be calculated and used to determine the absorption coefficients of the walls. A second approach is to record impulse responses in many points along two horizontal directions in the room while placing the microphone and loudspeaker by section 2.5. The amplitude of the corresponding frequency responses can then be analyzed at the room's resonance frequencies. The standing wave pattern revealed can then be used to determine the properties of a wall.

This chapter explains the measurement and post-processing methods used, including information about the three rooms where the measurements were taken and the measurement equipment. Furthermore, the signal processing performed on-site to help find the better source and receiver position for the measurements are explained.

3.1 Room description

The three rooms examined are all located on the same floor in the same building, and the rooms' boundaries are of similar construction. Common for all the rooms is that the concrete floor is covered in a wall-to-wall carpet.

Room A is a large conference room and is symmetrical along one axis. The two walls at $y = 0$ and $y = L_y$ are identical floor-to-ceiling glass walls. The laminated glass panes are 6.36 mm thick, 1.48 m wide, and either 0.6 or 1.97 m high. These glass panes are mounted in metal frames stretching from the floor to the suspended ceiling. There is also a wooden door with a hardened glass window mounted on each of these two walls. The wall at $x = L_x$ is made of concrete, with unknown thickness, stretching from the concrete floor to the concrete ceiling. The wall at $x = 0$ is a double-leafed light wall, which stops at the suspended ceiling.

(a) Room A **(b)** Room B **(c)** Room C

Figure 3.1: The three meeting rooms measured.

Each leaf comprises two 12.5 mm thick gypsum plates, with a 70 mm gap filled with mineral wool. The two wall leaves are connected with metal studs, places 60 cm apart. The suspended ceiling, in this case, is about 10 cm thick and is made up of a combination of mineral wool and gypsum board. The cavity between the suspended ceiling and the concrete ceiling above is isolated from neighboring rooms and hallways, with gypsum double-wall construction.

Room B is a smaller meeting room, as can be seen in Table 3.1. It has three walls made of gypsum system walls, and the remaining long wall, positioned along $y = 0$ is made up of big sheets of glass and a wooden door, like in-room A. The system walls are also double leafed with double gypsum plates on each side of the mineral wool-filled gap. The gap is 67.5 cm deep, and the metal studs are separated by 90 cm. The suspended ceiling in this room is only 40 mm thick, with a 1.25 cm thick gypsum plated glued on top. The cavity above is shared with the remainder of the fifth floor.

Room C is also a smaller meeting room, where all the boundaries have identical construction to room B. The wall at $y = 0$ is made of laminated glass sheets and a wooden door, and the opposing wall is made of only sheets of laminated glass. The two remaining walls are gypsum system walls of the same type as in room B.

Before measurement was conducted in any room, some preparatory work was done to assess the theoretical natural frequencies and optimal loudspeaker and source locations. Finally, the rooms were emptied of furniture before the measurements began. Inside room A, one table and chair remained with the operator in the room during these measurements. For rooms B and C, the operator left the room while recording the impulse responses.

Table 3.1: Room specifications.

Room	L_x	L_y	L_z	V	Ceiling
A	6.3 m	6.15 m	3.4 m	131.7 m^3	2.7 m
B	3.5 m	2.2 m	3.4 m	26.2 m^3	2.7 m
C	2.9 m	2.3 m	3.4 m	22.7 m^3	2.7 m

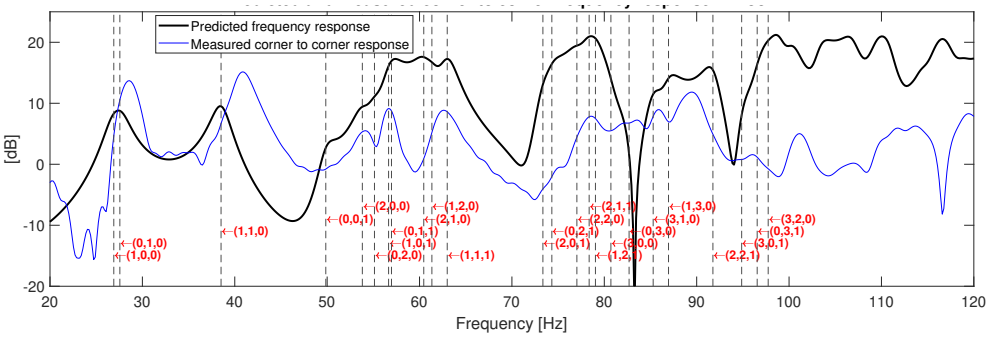


Figure 3.2: Predicted natural frequencies and frequency response compared to the measured corner to corner frequency response. The impact the loudspeaker has on the measured response have been reduced through division in the frequency domain.

3.2 Preparatory work

Some calculations were made to understand what room mode caused what resonance in the global frequency responses measured initially. The dimensions listed in Table 3.1 were used with Equation 2.12 and 2.13, to both predict the natural frequencies and the global frequency response of a rigid room with similar dimensions. Thus, the calculations were made with the source and receiver in a corner each. An example of such prediction compared to a corner to corner measurement can be seen in Figure 3.2.

The measurements done to measure the modal reverberation time in the room required a unique combination of microphone and loudspeaker position for each measurement taken. The calculated modal frequencies and the respective mode number, seen in Table 2.2 were used to examine modes adjacent to the one to be measured. That way, the microphone, and loudspeaker could be positioned such that these nearby modes were reduced substantially in amplitude. However, these positions in the table are based on the location of nodal planes in rigid rooms, and the actual locations had to be discovered experimentally in the actual room.

3.3 Measurements

3.3.1 Measurement setup

All measurements have been done using the Odeon measurement system, using an exponential sine sweep between 22 Hz and 16000 Hz. The input and output level differed between each series of measurements but was kept constant inside each series. Odeon does not keep the time delay before the direct sound from the source reaches the receiver and instead keeps the impulse response from a predetermined time before the direct sound arrives. This so-called silence before time was set to 0.1 seconds. The silence after time, which contains the tail of the impulse response and the background noise, was set to 3 seconds.

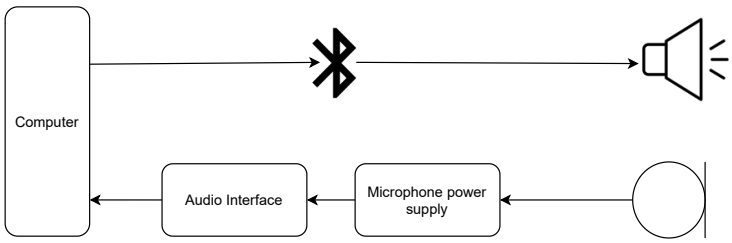


Figure 3.3: Block diagram of how the measurement equipment is connected.

The sound source used was a spherically radiating loudspeaker prototype from Odeon, connected directly to the laptop running the measurement software by Bluetooth. The loudspeaker can be seen in Figure 3.8, and its frequency response for low frequencies can be seen in Figure 3.9. A measurement microphone was then connected to the computer through an amplifier and an audio interface. The complete equipment list can be seen in Table A.1 and is illustrated in Figure 3.3.

3.3.2 Standing wave measurement

Initially, an impulse response with the source and the microphone each their corner was taken to obtain the room's global frequency response. The global frequency response was in turn used with calculation to identify the different room modes within the frequency response, as shown in Figure 3.2. To be able to isolate the lowest few axial modes in the x-direction, the loudspeaker were placed in $(L_x,L_y/2,L_z/2)$ to cancel modes with $n_x = 1$ or $n_z = 1$, which have been illustrated in Figure 2.3. As the nodal lines usually are not precisely where they would be in a room with rigid walls, the loudspeaker was moved slightly around this point, recording a series of impulse responses to find the position suppressing unwanted modes better. The microphone was then placed in $(0,L_y/4,L_z/4)$ to cancel the normal modes with $n_x = 2$ or $n_z = 2$. The microphone was also offset to the location where it best suppressed the unwanted normal modes.

After the best source and receiver position for the y- and z-dimensions had been determined, the microphone path had to be marked. A measurement tape

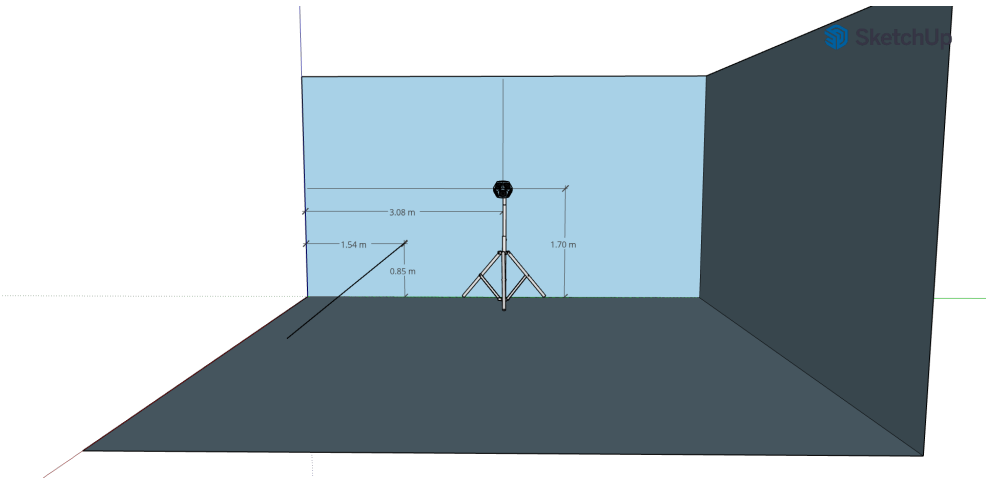


Figure 3.4: Illustration of loudspeaker and microphone positions in room A when measuring the standing wave patterns between the wall in x=0 and x=L_x. The microphone's acoustic center was moved along the black line.

was placed along the floor in the x-direction at the same y-coordinate as the microphone. While the microphone stand's height was kept constant, impulse response measurements were taken along the x-dimension. By moving the stand along with the measurement tape, it was possible to keep an even spacing, which has also been illustrated in Figure 3.4. Inside room A, which was bigger than the other rooms, an interval of 10 cm between each measurement was used. In the two other rooms, an interval of 5 cm was used. These measurements along the x-dimensions were then used to display the standing wave pattern to identify the nodal lines along the y-dimension to find the ideal source and receiver position along this dimension. The same measurement procedure was then performed along the y-dimension.

3.3.3 Modal reverberation time measurement

The standing wave patterns obtained from the two previous measurements were used to identify the location of nodal planes in both the x- and y-direction. The nodal planes in the z-direction were assumed equal to a rigid room, as the floor and ceiling were made of concrete. A few more measurements were taken to dial in on the source and receiver position giving better mode cancellation at those locations. Strings were then used to mark where the nodal planes intersected the floor. This were done along the experimentally found nodes equivalent to $L_x/2$, $L_x/4$ and $L_x/6$ and $L_y/2$, $L_y/4$ and $L_y/6$. These lines are illustrated in Figures 3.5, 3.6 and 3.7. The different intersections of the nodal planes mark optimal source and receiver positions that reduce the influence of respective modes in two orthogonal directions. Then, the loudspeaker and microphone stands were elevated to match a desired nodal plane perpendicular on the z-axis, like shown

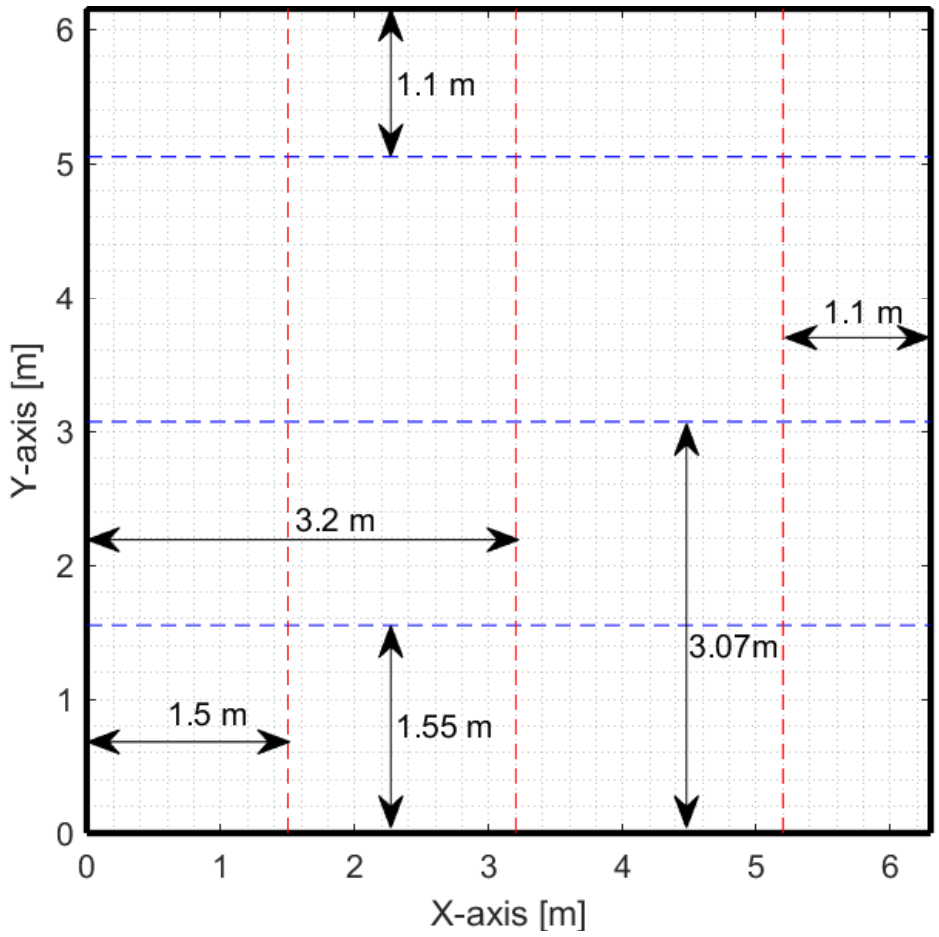


Figure 3.5: Pressure nodes discovered experimentally in room A.

in Figure 3.8. Thus, the acoustic center of the loudspeaker would usually be placed in either $L_z/2$ or $L_z/4$.

The source and receiver position could then be used to isolate specific room modes within a frequency response when combined with a bandpass filter. The source and receiver position can reduce the influence of modes that are close in frequency to the one wished to capture. The best location for the source and receiver has been decided by using Table 2.2 together with the actual nodal planes of the room. Different combinations of source and receiver positions have been used to record the impulse response of the lowest 2-3 axial modes in the x- and y-direction. For room B, only the first mode along the y-axis was recorded. The microphone and loudspeaker positions used to record these impulse responses are listed in Table A.2.

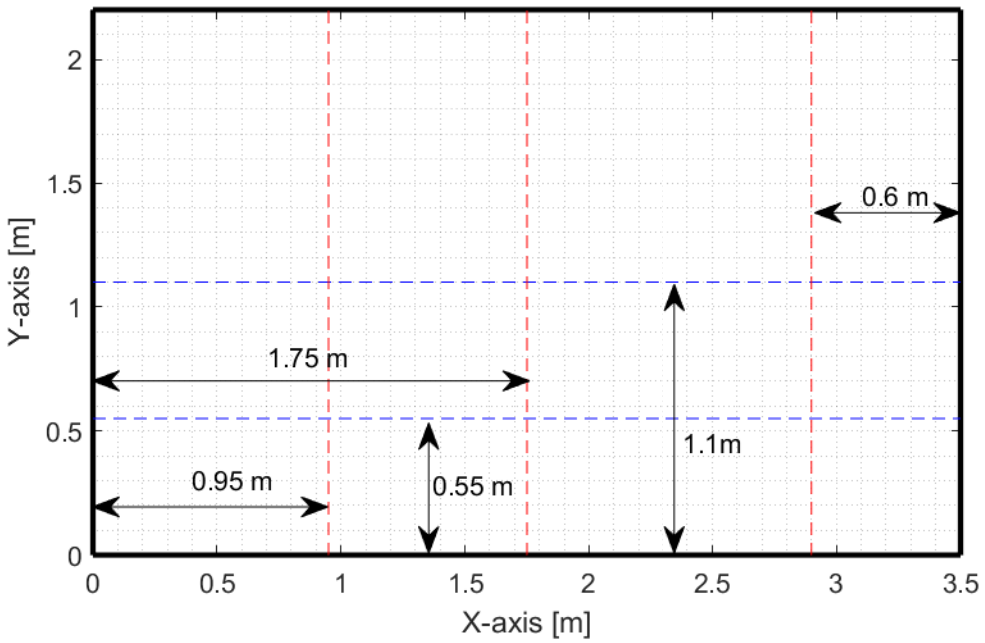


Figure 3.6: Pressure nodes discovered experimentally in room B.

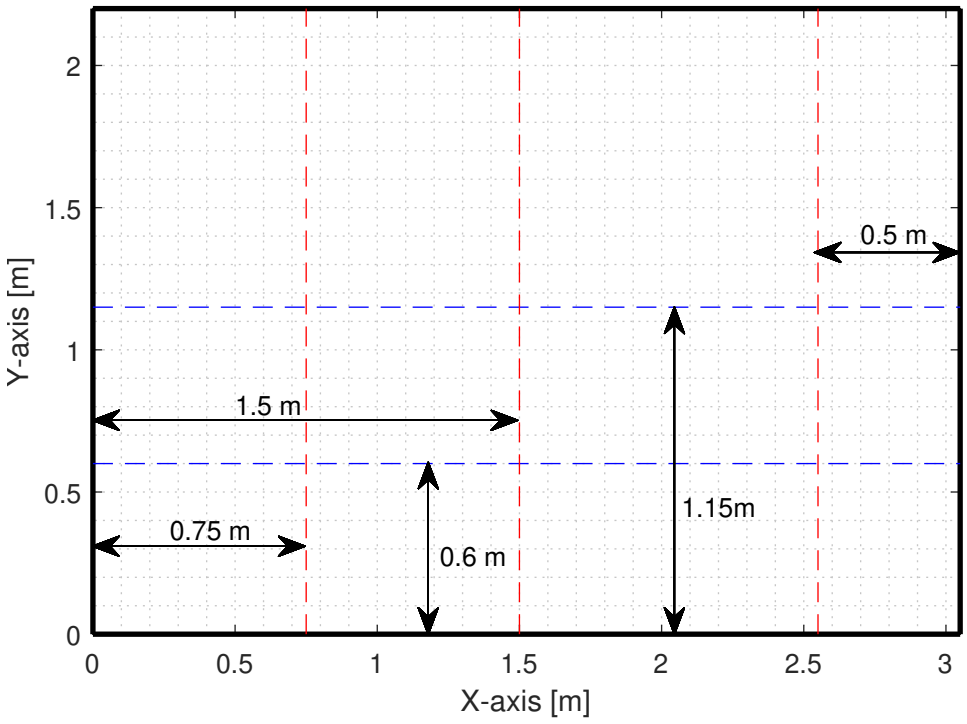


Figure 3.7: Pressure nodes discovered experimentally in room C.

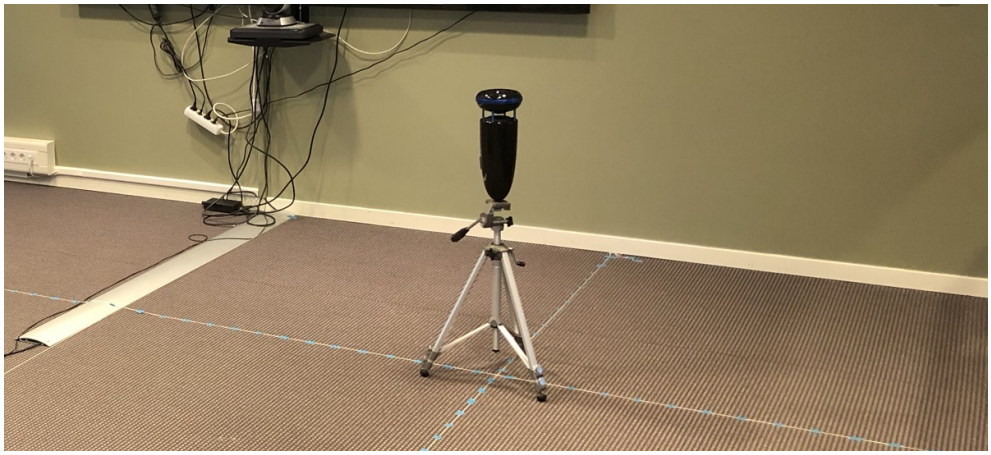


Figure 3.8: Source placed on an intersection between three nodal planes, as the acoustic center of the speaker is placed at a fourth of the room's height.

3.4 Post-processing

In order to import, process, do calculations and draw plots based on the impulse responses, Matlab has been the only tool used. There have been recorded between 110 and 130 impulse responses in the three rooms containing information about the standing wave pattern in two orthogonal directions in each room. There have also been recorded between 4 and 6 impulse responses to estimate the modal reverberation time. Before any results can be obtained, these impulse responses need to be read.

3.4.1 Treatment of impulse responses

As mentioned earlier, Odeon's measurement system does not keep any information about the time delay between the moment the loudspeaker emits the signal and when the microphone receives it. Also, all impulse responses are saved as wave files which means the signal is normalized to 1. However, Odeon saves an attenuation factor within the header of the file. As a wave file is written in a four-character code, the twelfth set of four bytes contains the attenuation factor. After an impulse response has been divided by this factor, the original amplitude of the signal is recovered.

After the original amplitude of the impulse response has been recovered, it can be transformed to the frequency domain. Firstly, the impulse response is truncated at 0.8 times the reverberation time in the room. Odeon has calculated the reverberation time from all the impulse responses recorded in each room. Ideally, a truncation would happen 10 dB above the noise floor in the impulse response, but manually inspecting several hundred impulse responses is not feasible. Thus this simplified way of deciding where to truncate the impulse response has been used. A discrete Fourier transform is then done using Matlab's fast Fourier trans-

form (FFT) algorithm. The FFT size has been set to 2^18 to obtain a high-resolution frequency response in the lower frequencies.

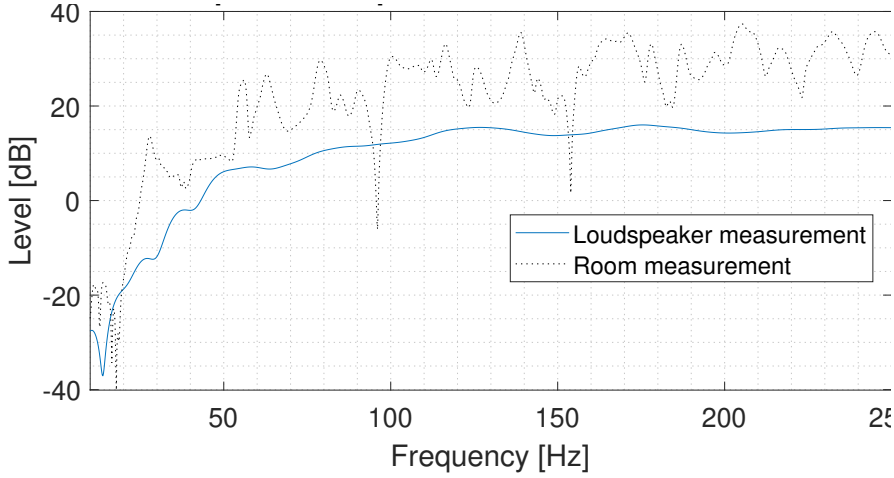


Figure 3.9: Frequency response of the loudspeaker and an arbitrary room, showing the effect the loudspeaker response have on the room's frequency response.

Finally, the loudspeaker's effect on the frequency response has been reduced. Odeon provided an averaged impulse response of the loudspeaker, recorded in an anechoic chamber to make this possible. The loudspeaker's impulse response has been truncated after 7000 samples, and Fourier transformed the same way as the room's impulse response. The room's frequency response is then divided with the loudspeakers frequency response, and thus the influence from the loudspeaker on the measurements has been reduced.

3.4.2 Isolating single room modes

During the measurements, the source and receiver positions were used to reduce room modes nearby the room mode, which was to be measured. The procedure described above has been used to obtain the frequency responses of these measurements. Although the nearby modes have been reduced, modes further away are still present. Bandpass filters were used to counter this. The bandpass filters' center frequency was set to a frequency close to the peak of the mode wanted isolated. The relation in Equation 2.24 was attempted fulfilled when picking filter bandwidth to avoid introducing reverberation times from the filters. The backward filtering technique described in section 2.8 was also used. In Figure 3.10, an example frequency response before a bandpass filter is applied can be seen, as well as the frequency response of the bandpass filter that is to be applied.

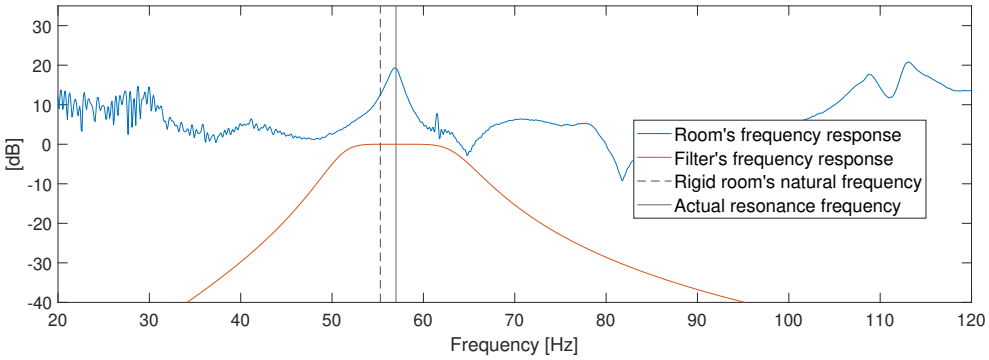


Figure 3.10: Example of how the frequency response of a measurement of a single mode looks, and the frequency response of the filter designed to completely isolate it.

Chapter 4

Results

In this chapter, both the intermediate and final results are presented, including the global frequency response in the three rooms, where all the room resonances should be enabled. The global frequency responses are presented with the predicted global frequency response and predicted location of the normal frequencies in the room. The frequency responses obtained from both the standing wave measurement and the modal reverberation time are also shown. Here the frequency response of the bandpass filters used with the modal reverberation time measurements also is displayed.

Then, the standing wave pattern and a least-squares-fit model of the measured pattern are created and plotted for the first two or three axial room modes in two horizontal directions in each room. The standing wave patterns have been used to calculate the respective surfaces' complex reflection factor and absorption coefficient. The absorption coefficient has also been calculated from the modal reverberation time, and the decay curve of the axial room modes is presented together with the respective reverberation time and absorption coefficient. Finally, the predicted absorption coefficient for a single wall with the same dimension and materials as the wall segments in the real rooms is also shown.

The theory and methods used to obtain these results are also described, with references to the theory and method chapter. The results will be compared and evaluated in the next chapter.

4.1 Global frequency responses

The global frequency responses were obtained through the corner to corner measurements in the three different rooms. The frequency response of the room after dividing by the frequency response of the loudspeaker is shown in Figures 4.1-4.3. The predicted frequency response for an equivalent-sized cuboid room with rigid walls has been calculated from Equation 2.13, by inserting the room's dimensions and reverberation time. In this case, the room's reverberation time at the 125 Hz octave band, found in section A.3, has been used. The natural frequencies estim-

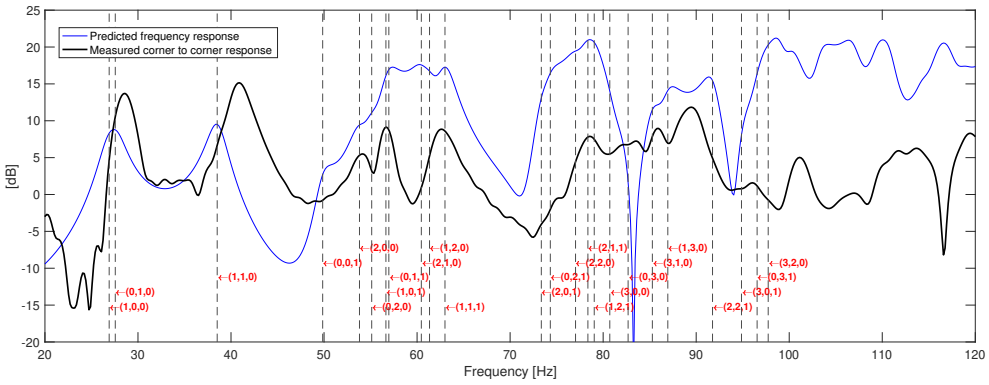


Figure 4.1: Global frequency response measured and predicted frequency response in room A, measured from corner to corner.

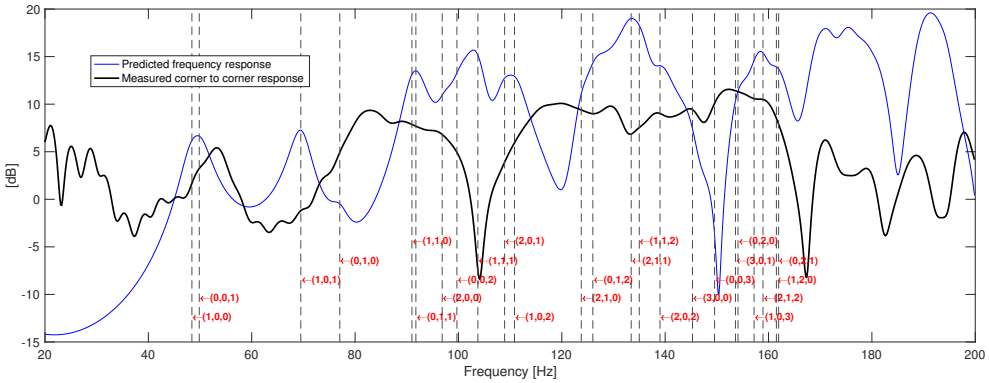


Figure 4.2: Global frequency response measured and predicted frequency response in room B, measured from corner to corner.

ated for the rigid room have also been calculated through Equation 2.12 and are displayed as vertical black dashed lines, with the mode numbers attached.

4.2 Standing wave ratio methods

4.2.1 Transfer functions

The frequency response of each microphone position taken along one of the room's dimensions is shown in Figures 4.4-4.6. The loudspeaker's influence has been removed from all the measurements, and the original amplitude has been restored as explained in subsection 3.4.1. Each plot includes frequency responses from all points measured along the same room axis. The x-axis corresponds to the room's length, and the y-axis corresponds to the room's width. The microphone and loudspeaker positions are thoroughly described in subsection 3.3.2. When comparing Figures 4.1-4.3 to Figures 4.4-4.6, it can be seen that most room resonances in the low frequencies have been suppressed, and mostly the axial room modes are

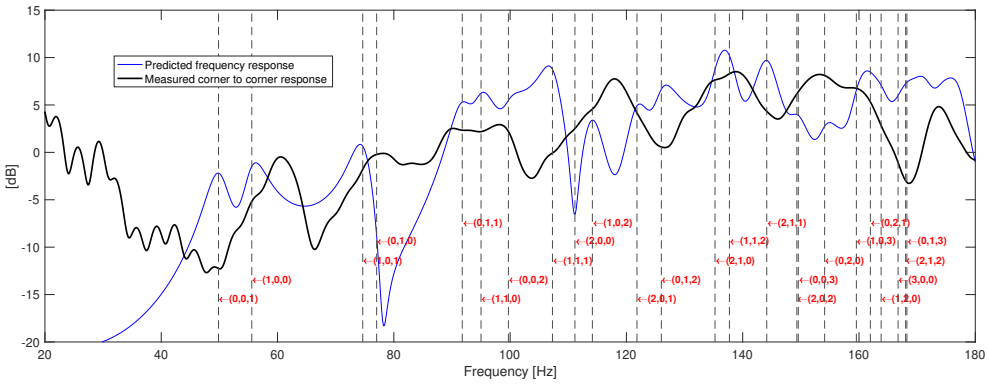


Figure 4.3: Global frequency response measured and predicted frequency response in room C, measured from corner to corner.

left.

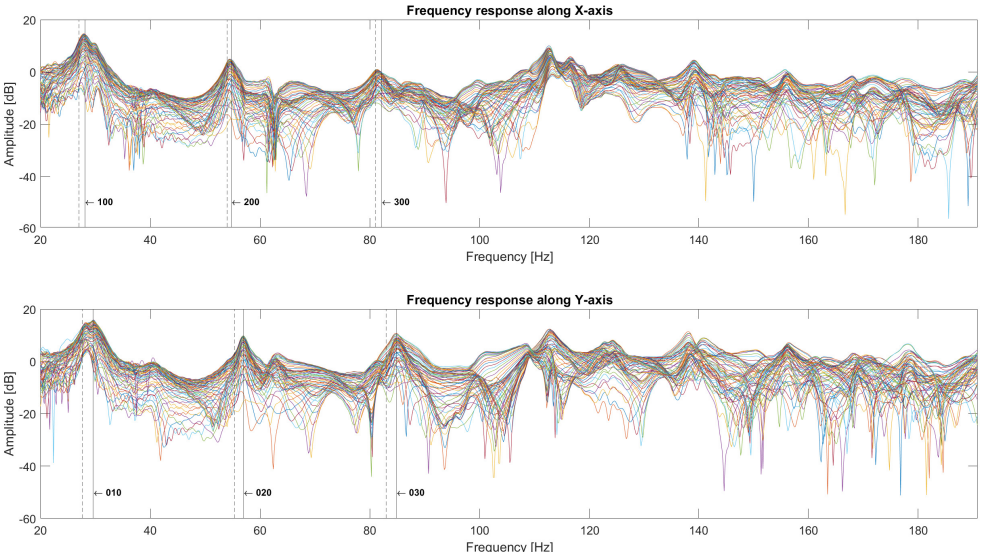


Figure 4.4: Frequency response along X- and Y-axis of room A. The black dashed lines marks theoretical natural frequencies with rigid surfaces and the black solid line marks their actual locations in the real room.

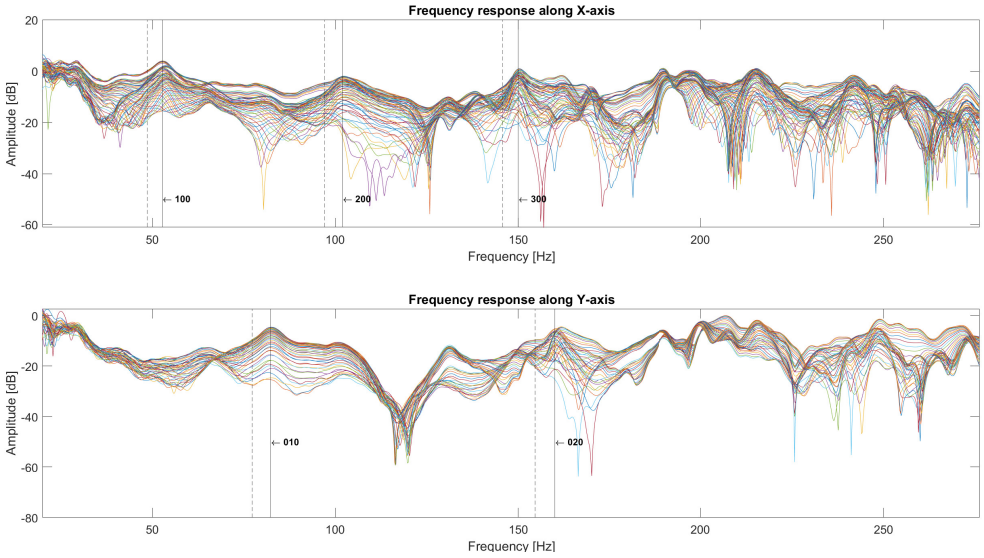


Figure 4.5: Frequency response along X- and Y-axis of room B.

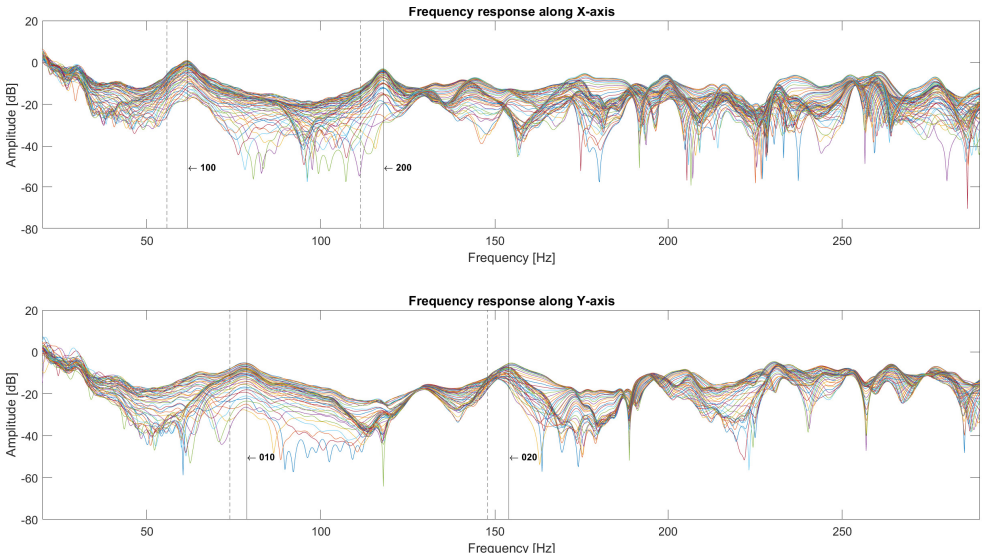


Figure 4.6: Frequency response along X- an Y-axis of room C.

4.2.2 Standing wave ratio

The standing wave pattern is obtained by inspecting the measured frequency responses, in Figures 4.4-4.6, at one of the room's resonance frequencies. The center frequencies of these resonances are easily found by inspecting the frequency responses shown in Figures 4.4-4.6, as indicated by solid vertical lines. The standing wave pattern is shown in Figures 4.7-4.14, together with the least-squares-fit model. The least-squares-fit model is calculated through the method described in section 2.7 by evaluating the values of the points marked with red circles in Figures 4.7-4.14. It can be seen in Figure 4.9 that the sound pressure close to the walls deviates from an interference pattern caused by two plane waves traveling in the opposite direction of each other. Thus the points closest to the boundaries are left out when calculating the least-squares-fit model.

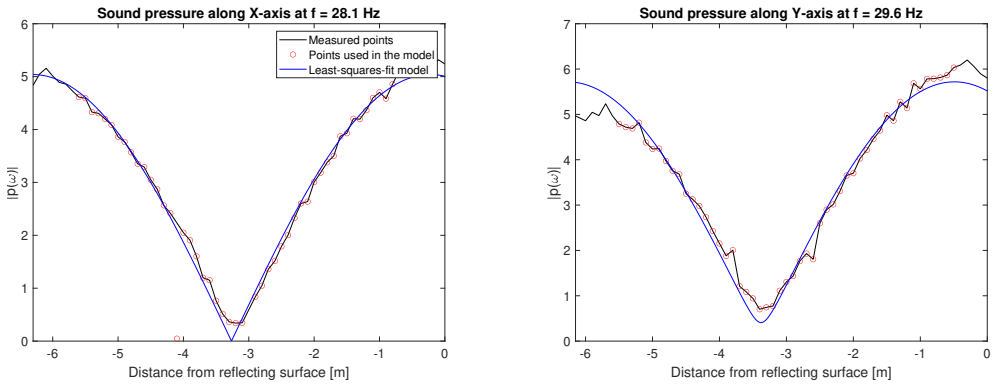


Figure 4.7: Measured and modelled standing wave, mode (1,0,0) and (0,1,0) in room A.

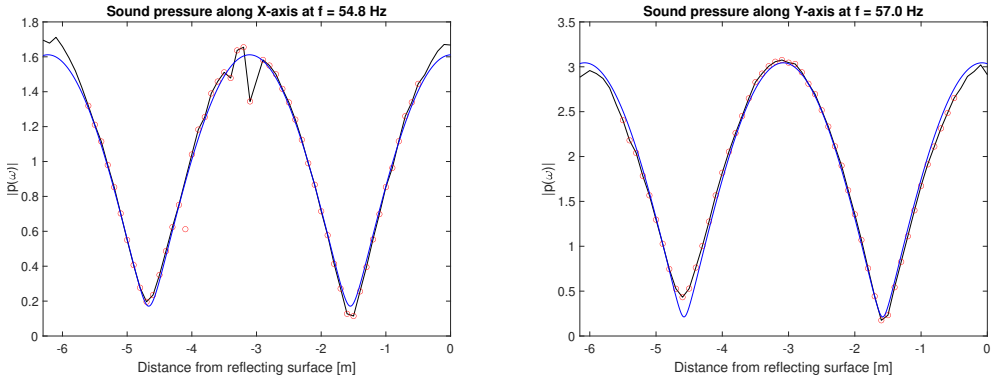


Figure 4.8: Measured and modelled standing wave, mode (2,0,0) and (0,2,0) in room A.

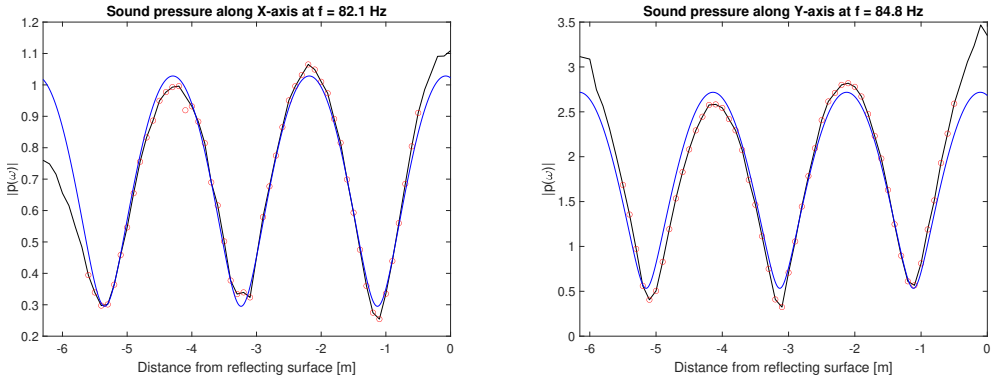


Figure 4.9: Measured and modelled standing wave, mode (3,0,0) and (0,3,0) in room A.

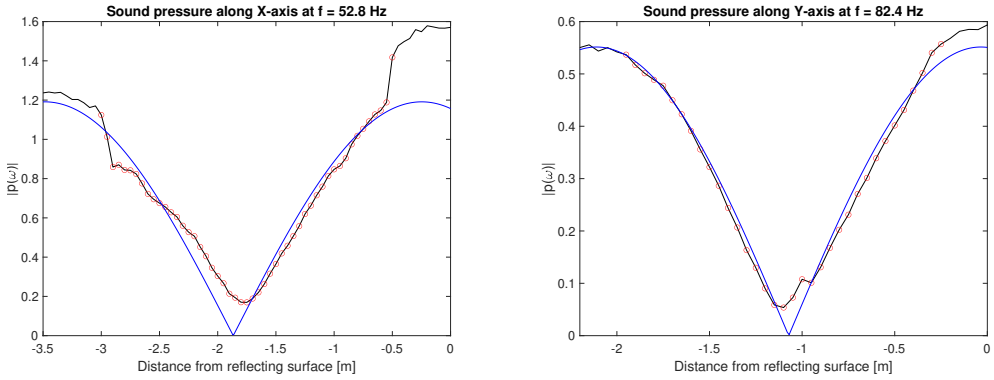


Figure 4.10: Measured and modelled standing wave, mode (1,0,0) and (0,1,0) in room B.

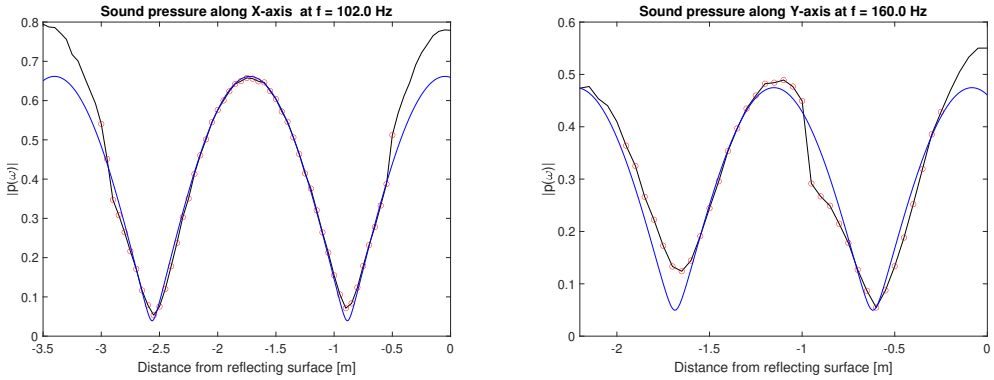


Figure 4.11: Measured and modelled standing wave, mode (2,0,0) and (0,2,0) in room B.

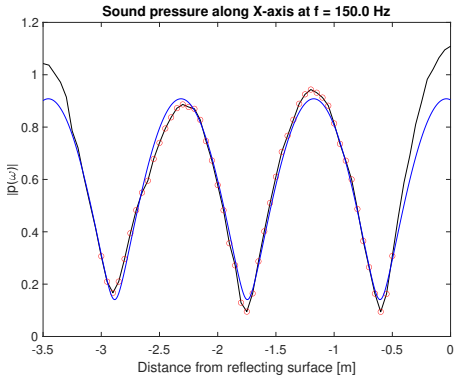


Figure 4.12: Measured and modelled standing wave, mode (3,0,0) in room B.

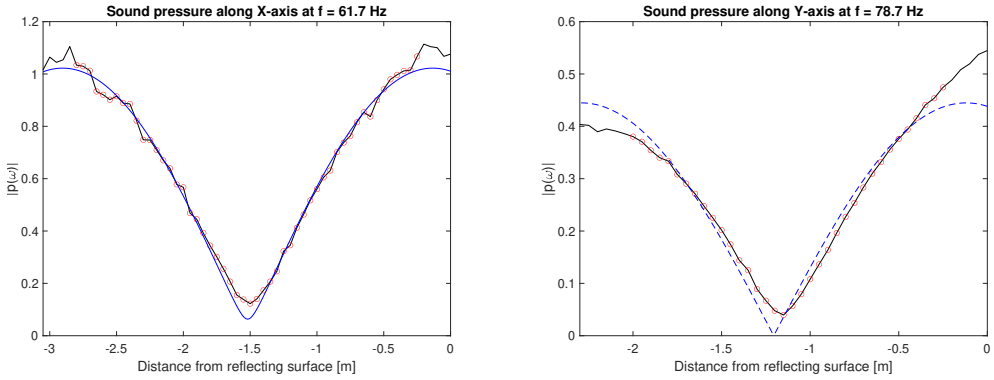


Figure 4.13: Measured and modelled standing wave, mode (1,0,0) and (0,1,0) in room C.

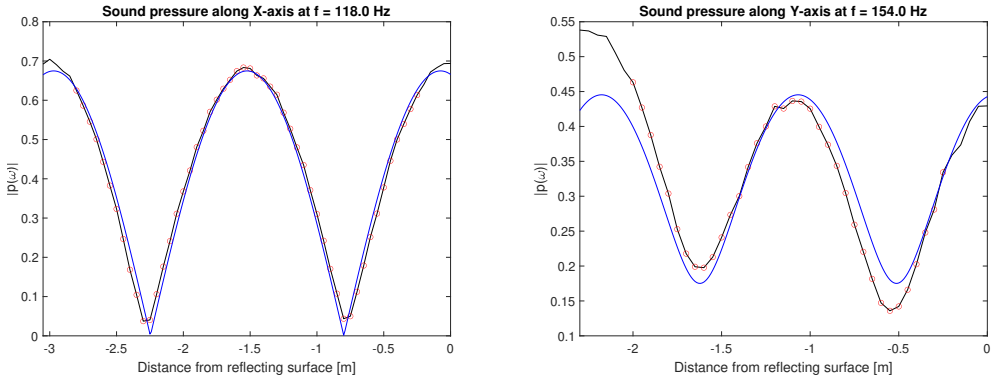


Figure 4.14: Measured and modelled standing wave, mode (2,0,0) and (0,2,0) in room C.

4.3 Modal reverberation time method

4.3.1 Transfer functions

The frequency responses of the measurements, described in subsection 3.3.3, is shown in the Figures 4.15-4.17. The loudspeaker's response has been divided from all the recorded frequency responses. In addition, the natural frequency for the equivalent rigid room, the actual resonance frequency, and the frequency response of the bandpass filter isolating the room mode is shown too. The natural frequencies of an equal-sized rigid room are calculated through Equation 2.12, and the actual ones have been found through inspecting the frequency responses recorded. The filter's frequency response was obtained by running a unit pulse through the filter and taking a DFT of the output. As described in section 2.8, the bandwidth of the bandpass filter has been attempted to be as narrow as needed while not distorting the impulse responses of the room modes. The bandpass filter's center frequencies have also been offset to the mode's center frequency in some cases to avoid interference from other modes.

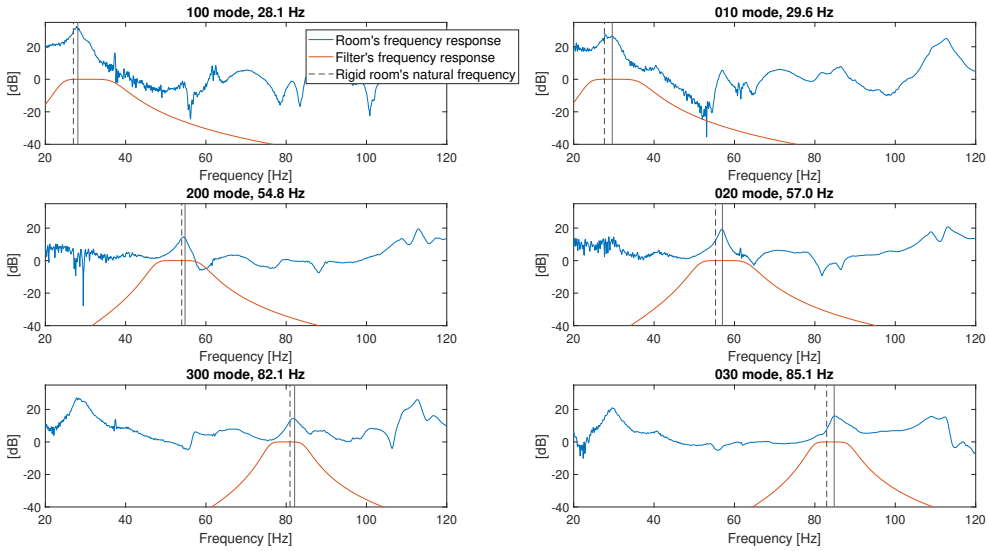


Figure 4.15: Frequency responses of modes in room A. The black dashed lines marks theoretical natural frequencies with rigid surfaces and the black solid lines marks their actual locations in the real room.

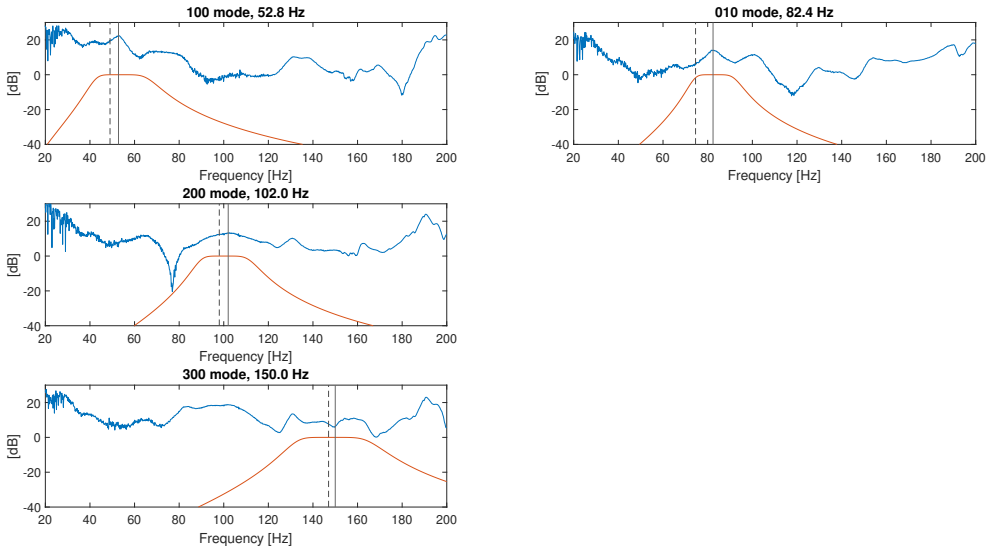


Figure 4.16: Frequency responses of modes in room B.

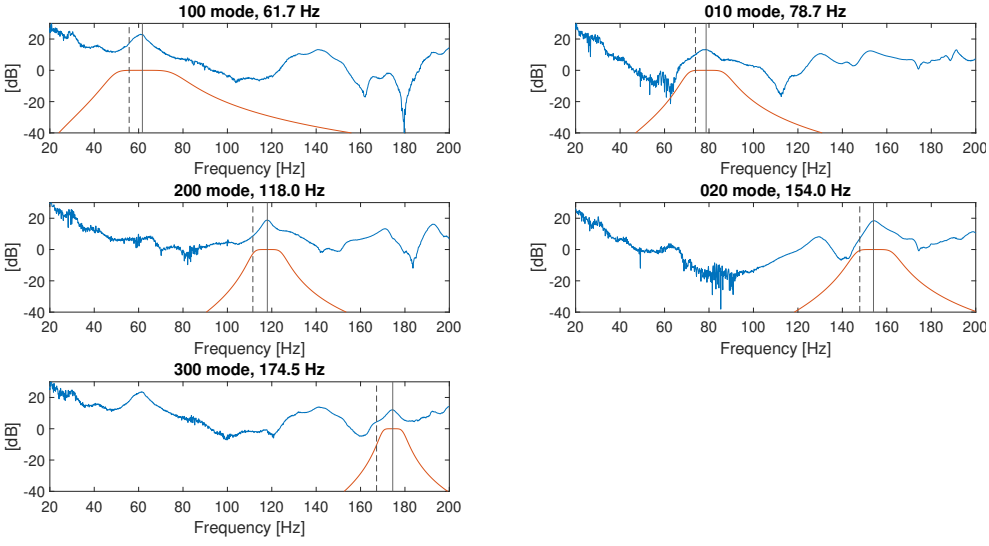


Figure 4.17: Frequency responses of modes in room C.

4.3.2 Decay curves and absorption coefficient of axial room modes

A straight decay curve is needed to obtain a modal reverberation time. A decay curve of a single room mode is obtained by backward integration of the band-pass filtered impulse responses. The frequency response of the signal before filtering can be seen in Figures 4.15-4.17 together with the frequency response of the bandpass filter used on the impulse response. As the source and receiver positions reduce the influence of other normal modes in the immediate vicinity and the bandpass filter reduces the influence of more distant normal modes, this impulse response will ideally contain information from a single normal mode. How the room mode is isolated is also illustrated in Figure 2.4. The decay curve of the different normal modes in the three rooms is shown in Figures 4.18-4.20.

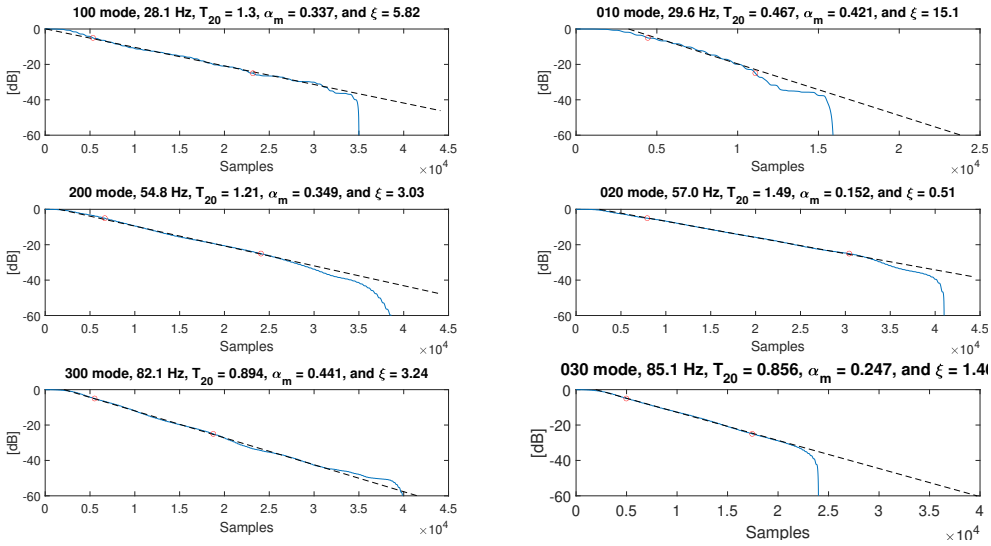


Figure 4.18: Decay curve, modal reverberation time, mean absorption coefficient and linearity ratio of normal modes in room A.

As the decay curve alone is prone to the truncation effect, a straight least-squares-fit line is calculated. All values between the two red circles marked on the decay curves are used to calculate the coefficients for the regression line. This region is also known as the T_{20} evaluation range between -5 and -25 dB compared to the initial value of the decay curve.

The time it takes this least-squares-fit line, the black dashed line, to go from 0 to -60 dB is then the modal reverberation time. The degree of non-linearity is then calculated through Equation 2.26, and would ideally be less than 10‰. At last, the mean absorption coefficient of the two parallel walls, which the respective normal mode interacts with, is obtained through Equation 2.28.

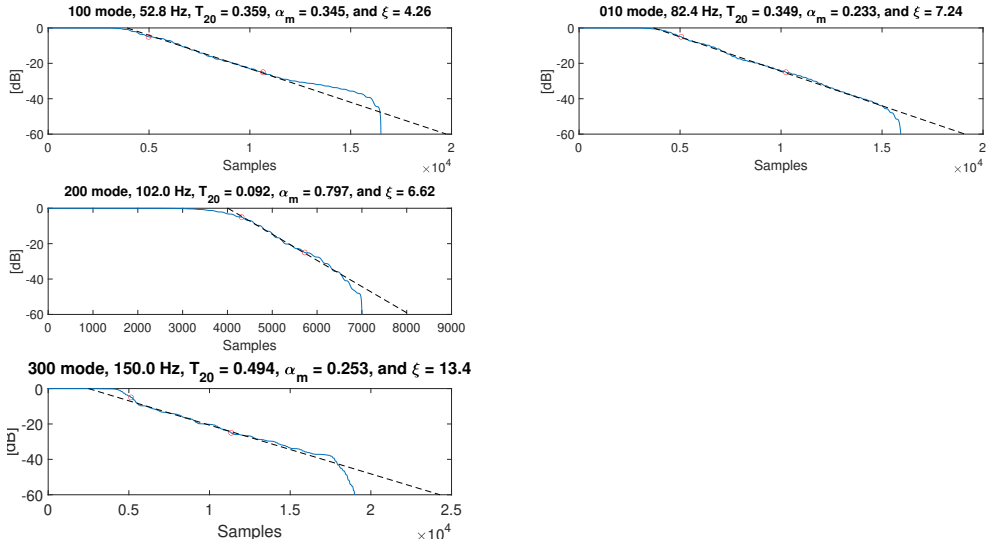


Figure 4.19: Decay curve, modal reverberation time, mean absorption coefficient and linearity ratio of normal modes in room B.

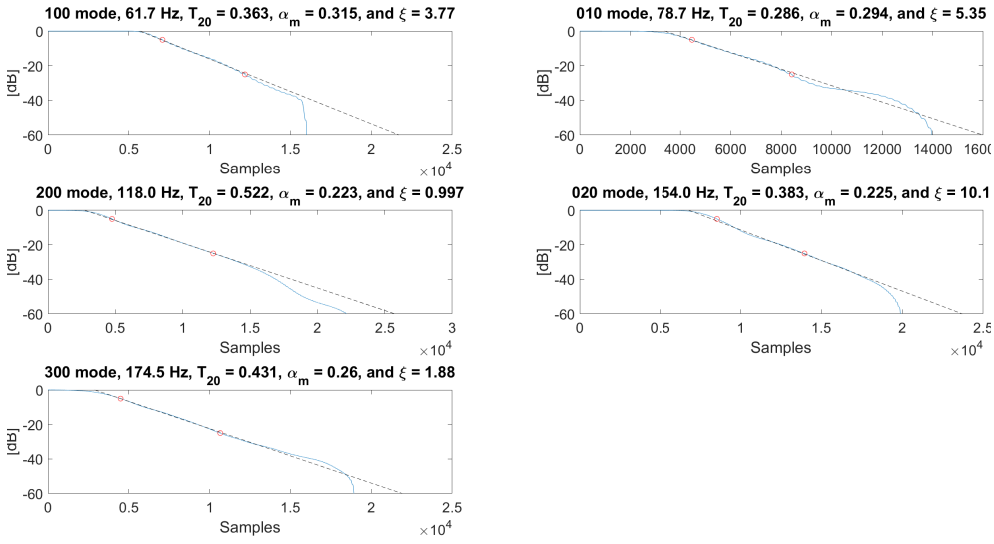


Figure 4.20: Decay curve, modal reverberation time, mean absorption coefficient and linearity ratio of normal modes in room C.

4.4 Comparison of absorption coefficients

The standing wave ratio is used to determine the wall impedance, reflection factor, and absorption coefficient. The first pressure minimum and first pressure maximum when moving away from the wall at x=0 or y=0 is used to calculate the standing wave ratio s, through Equation 2.14. The first maximum is close to the opposite wall in the special case of the (1,0,0) and (0,1,0) modes. Thus, the pressure maximum close to the opposite wall is used. Then to estimate the speed of sound in the rooms, Equation 2.17 is used with the location of the first two pressure minima in Figure 4.9, moving away from the wall in x or y equal to 0. The speed of sound using the measured data is $c = 339$ m/s, and the speed of sound through the least-squares-fit model is $c = 341$ m/s. The modulus and phase angle of the reflection coefficient can then be calculated through Equation 2.15 and 2.16 respectively. The modulus of the reflection factor can be directly used with Equation 2.3 to obtain the absorption coefficient. After converting the reflection factor from polar to complex numbers, the specific impedance of the wall can be obtained through Equation 2.1. The standing wave ratio, reflection factor, absorption coefficient, and characteristic wall impedance of different axial modes in the three rooms can be seen in Table A.3 based on the measurements, and in Table A.4 based off the least-squares-fit model.

At last, some quality criteria have to be set in both cases. The criteria for the standing wave method are that the measured standing wave has to look like an interference pattern between two plane waves. The absorption coefficients in Figure 4.21, marked by circles, are from the standing wave method, and the points that meet the criteria are enlarged. For the least-squares-fit model, marked with plus symbols, no data points have been enlarged due to the general deviation from all other estimation methods.

The absorption coefficients calculated through modal reverberation time must meet two quality criteria. The first is the degree of non-linearity, ξ, which must be less than 10‰. The second is that the product of the filter bandwidth and reverberation time must be more than four. The values for these parameters can be seen in Table 4.1, as well as if they meet all criteria, denoted by OK, or denoted by the criteria they break if that is the case.

Table 4.1: Quality criterias for modal reverberation time.

Room	A						B			C				
Mode	100	200	300	010	020	030	100	200	300	100	200	300	010	020
ξ [‰]	5.82	3.03	3.24	15.1	0.51	1.46	4.26	6.62	13.4	3.77	0.997	1.88	5.35	10.1
B [Hz]	14.03	12.23	9.25	13.8	13.2	9.71	24.76	23.16	34.27	28.51	13.63	10.1	18.15	17.82
T [s]	1.3	1.21	0.894	0.467	1.49	0.856	0.359	0.092	0.494	0.363	0.522	0.431	0.286	0.383
$B \cdot T$	18.24	14.80	8.27	6.44	19.67	8.31	8.89	2.13	16.93	10.35	7.11	4.35	5.19	6.83
Verdict	OK	OK	OK	ξ	OK	OK	OK	$B \cdot T$	ξ	OK	OK	OK	OK	ξ

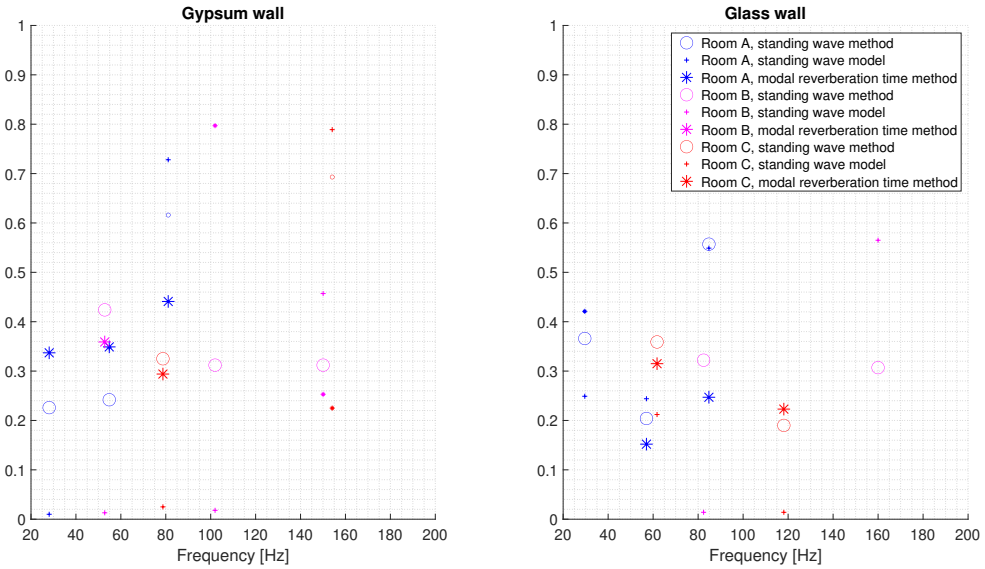


Figure 4.21: Absorption coefficients estimated through the different methods. Data points that meets the quality criteria are enlarged.

4.5 Predicted absorption coefficients

Predictions of the wall properties have been made to have ground to evaluate the measured absorption coefficient. The predictions carry huge uncertainties due to simplifications, and variable values assumed. The first simplification to the problem is treating the gypsum walls as single-leaf walls instead of double-leaf with a mineral wool-filled cavity. Then only the wall segment between two studs will be considered. This segment corresponds to a 25 mm thick, 60 cm wide, and 270 cm tall wall segment for room A. For rooms B and C, the width is 90 cm instead due to the longer stud distance of the system wall.

As details about the wall constructions are known, the different wall segments' critical frequency and fundamental structural resonance frequency can be estimated. By using data from Table 2.1, which can be used to calculate the critical frequency through (2.4). As the gypsum wall segment is 25 mm thick but made of two 12.5 mm gypsum plates that are not glued together, the critical frequency is calculated for a 12.5 mm thick plate. The laminated glass sheets are 8.36 mm thick for room A and 6.36 mm thick for rooms B and C. Then, the frequency of the fundamental structural mode can be calculated for each plate. The dimension of the wall segment and its critical frequency is used to find the fundamental frequency through Equation 2.5. These sizes of each segment can be seen in are described in section 3.1. The wall segments close to the wall boundary, which could be shorter, have been neglected.

Finally, the wall impedance and absorption coefficients can be estimated. However, some variables have been set to reasonable values, as either not enough information is available to calculate the real values or the process is too com-

plicated. This simplification is again one of the reasons that these estimates will carry significant uncertainties. The first of these values is the structural absorption coefficients, α_s, which will be set to 0.1 for all wall segments. The second is the resonant radiation efficiency, σ_{res} which has been set to 1.

The total loss factor is then calculated through Equation 2.7, where the initial loss factor η_{int} for each material was found through Table 2.1, the perimeter and surface area of the plates have been calculated from the plate dimensions, and the critical frequency was calculated earlier in this section. The total loss factor, mass, and fundamental structural frequency of the wall segment can then be inserted into Equation 2.6 to estimate the wall impedance. The normalized specific impedance, $\zeta = Z/\rho_0 c_0$ can in turn be used with Equation 2.2 and 2.3 to obtain the absorption coefficient. The absorption coefficients for the different wall segments can be seen in Figure 4.22-4.26.

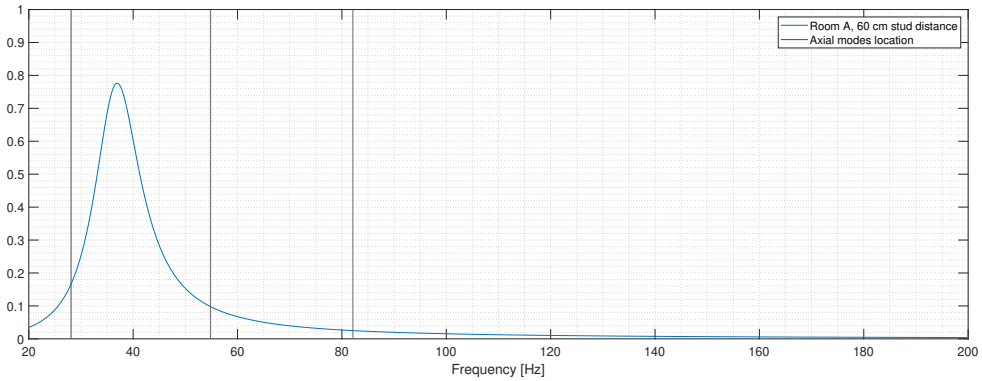


Figure 4.22: Estimation of absorption coefficient for 25 mm thick, 60 cm wide and 270 cm tall gypsum wall segment in room A.

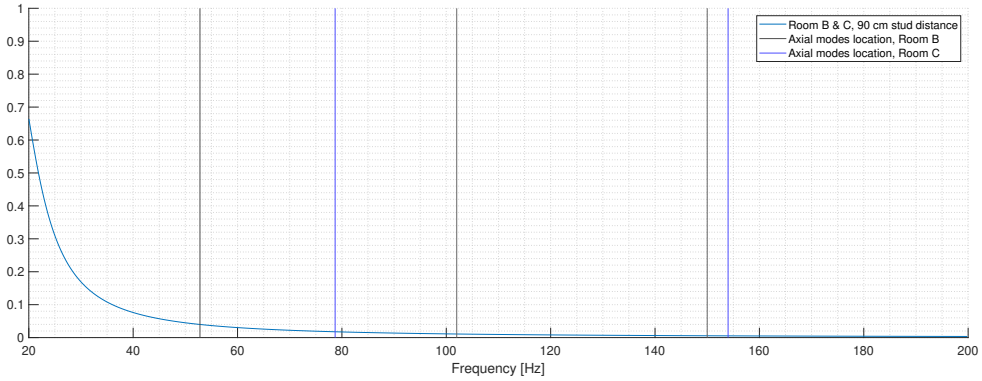


Figure 4.23: Estimation of absorption coefficient for 25 mm thick, 90 cm wide and 270 cm tall gypsum wall segment in rooms B and C.

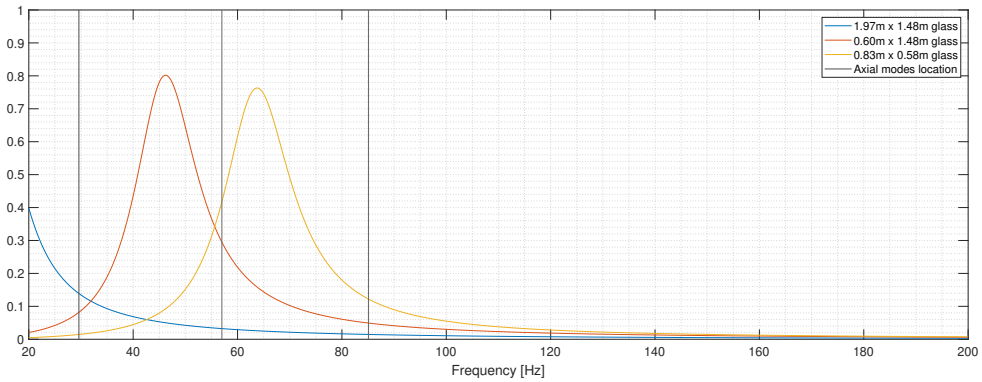


Figure 4.24: Estimation of absorption coefficient for 8.36 mm thick glass wall segment in room A.

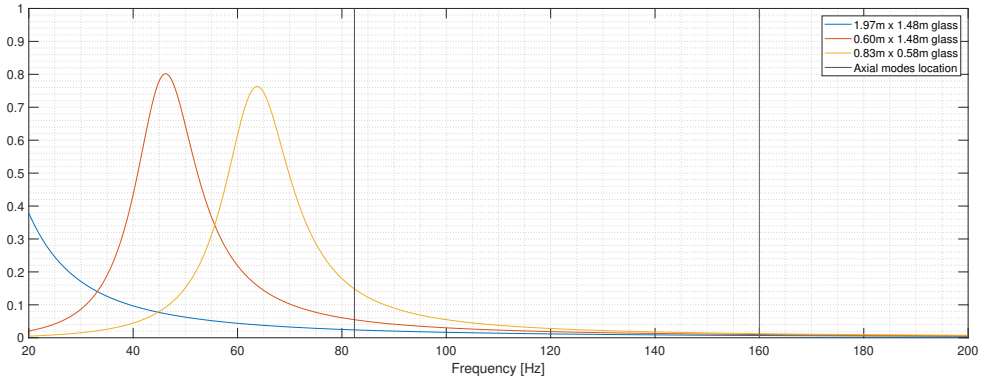


Figure 4.25: Estimation of absorption coefficient for 6.36 mm thick glass wall segment in room B.

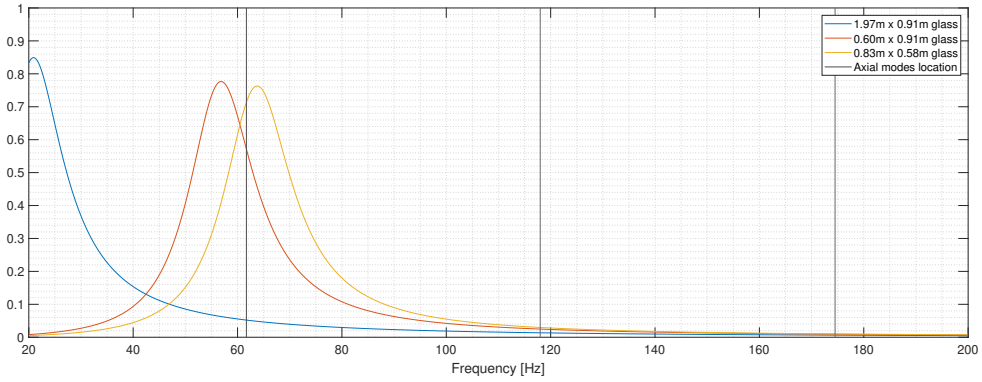


Figure 4.26: Estimation of absorption coefficient for 6.36 mm thick glass wall segment in room C.

Chapter 5

Discussion

5.1 Measurements

5.1.1 Equipment and software

Most of the equipment used has been relatively standard, except for the loudspeaker. The loudspeaker, a prototype of an omnidirectional loudspeaker in development at Odeon, has had a significantly smooth frequency response within the frequency range used. As can be seen in Figure 3.9, it is smooth down to about 50 Hz before it starts rolling off. Its small size has made it relatively easy to accurately place the loudspeaker, which has been helpful when pressure nodes have been located experimentally. The downside is that the loudspeaker is wireless and connected via Bluetooth technology to the measurement system. Although this causes no problems measuring most room acoustic parameters, recording the time delay between the source and receiver is impossible.

Odeon measurement systems have been used to record and save impulse responses. This software is great for statistical analysis of multiple impulse responses and calculating room acoustic parameters like reverberation time, early decay time, speech transmission index, and so on. Although it only supports saving impulse responses as wave files, which normalizes the amplitude of the signal, it counteracts this by saving the attenuation factor into the header of the sound file so that the amplitude can be restored. On the other hand, the software does not keep information about the time delay between the source and receiver. It removes the beginning of the impulse response and only keeps the impulse response from a predetermined time before the direct sound. Thus the phase information of the signal is lost, and only the magnitude of the transfer function is available. These complications with the time delay caused by the loudspeaker and software have excluded specific post-processing methods, which will be discussed later. Another challenge the measurement software provides is navigating the impulse response on-site. It is not easy to zoom in and out and navigate around the impulse response recorded. Thus, it becomes much more difficult quality checking the impulse responses, e.g., making sure the direct sound is well defined after recording

a measurement.

5.1.2 Measurement method

Besides the challenges imposed by the measurement system, the type of rooms used has certainly not been ideal. In-situ low-frequency measurements of absorption coefficient and wall impedance is a fairly unexplored area, and although many real rooms have suspended ceilings, it imposes uncertainties. It becomes even more challenging when the space above the suspended ceilings is shared with other rooms. The suspended ceiling is most likely effectively transparent at very low frequencies but might not be for higher frequencies. As can be seen in Figures 4.1-4.3, the (0,0,1) mode is heavily damped and cannot be located, and the (0,0,2) mode is not possible to identify either. Thus, even if the virtual height of the room would change at a frequency, the vertical modes are heavily damped and have not been expected to impact the results much. However, the significance of complications caused by the common volume above the suspended ceiling in rooms B and C are unknown.

Executing the measurements went without any significant complications. As mentioned, the loudspeaker had a small footprint compared to other omnidirectional loudspeakers, which made it easier to place it close to the actual pressure nodes. The most time-efficient way of doing the measurement was to start recording impulse responses along a line normal to two opposing walls of the same material, as explained in subsection 3.3.2. This way, the first node line would be in the dead center, and the loudspeaker placement went quite quickly. The microphone placement still took a few attempts to optimize for each room, as this nodal line often had been shifted ever so slightly compared to the calculations based on rigid walls. Then the standing wave patterns at the first three axial modes obtained by this measurement could be used to reveal the nodal lines perpendicular to the ones already used. As the last two walls did not have the same construction, this reduced the time used to locate these pressure nodes drastically. After standing wave patterns were obtained in both the x- and y-directions, these could be used to identify the nodal lines shown in **??**. Thus, initially revealing the standing wave pattern of the first axial modes helped the modal reverberation time measurements to go quickly.

5.2 Standing wave ratio method

The method of measuring the absorption coefficient of a wall, through the standing wave pattern of plane waves in the room, has been to some extent adapted from the impedance tube method [5]. The key differences are that a room is used instead with mode canceling through the transducer positions, a stationary microphone instead of a moving one, and using a logarithmic sweep signal instead of a sinusoidal signal with a constant frequency. Together these differences make it much more bothersome to locate the exact positions of the pressure minima.

Other modes within the same frequency range must be reduced sufficiently in amplitude, and the sound field recorded must behave as a plane wave for the axial modes examined to get results that can be assumed reliable. Furthermore, the resolution of the measurements must be high enough to capture the extrema of the standing wave pattern. Also, this measurement method was not intended for low frequencies, and in the case of fundamental modes like (0,1,0), the first pressure maximum when moving away from the wall that is attempted measured is close to the opposing wall. As seen in several cases, the pressure amplitude is often raised close to the walls, affecting the value for the absorption coefficient. The impact the increased amplitude close to the walls may have on the results is discussed later on.

By comparing the frequency responses in Figure 4.4-4.6 with the global frequency response of the room in Figure 4.1 it is possible to see to what degree the unwanted modes have been reduced. The three first axial modes in room A are the only three prominent peaks below 100 Hz, and most other modes are sufficiently reduced. In rooms B and C, which are smaller rooms, there are other prominent peaks from the frequency responses measured along the y-axis of the rooms. Another thing to notice is the small peak to the left of the (0,1,0) mode in Figure 4.5. This peak might indicate that the suspended ceiling acts as the room boundary already at this frequency, and it is the (0,0,1) mode observed. If the suspended ceiling is reflective at this frequency, the loudspeaker is positioned too high to suppress the (0,0,1) mode. However, the axial modes in the z-direction are substantially damped and do not cause trouble by themselves. On the other hand, tangential modes might be an issue. It may be tangential modes that appear as peaks in between the first three axial modes in Figure 4.5 and 4.6, which may cause unwanted energy at the frequency of the axial modes that are being examined.

Another thing to notice is that the third axial mode has become harder to capture as the room size has decreased. The (3,0,0) mode was visible in room B, where the room's length is 3.5 meters, while this mode in room C was not recognizable. The (0,3,0) mode has not been captured in both the smaller rooms, as it could not be found by inspecting the wave pattern between the walls.

Furthermore, it becomes apparent that the axial modes in Figure 4.4-4.5 are not harmonics of each other as one would expect from Equation 2.12. The dashed lines represent the theoretical natural frequencies if the walls had rigid boundaries, and it seems like the measured resonances in the room are consistently higher. This frequency shift is likely caused by a complex impedance on the wall where the reactance of the wall imposes a phase shift, virtually moving the room boundary into the room, which increases the natural frequency of the respective room mode. This frequency error is more significant for the lowest frequency and is expected to become negligible for higher frequencies where the wall would act more like rigid boundaries. The fact that the frequency spacing between the lowest axial modes varies further emphasizes how the theory for rigid boundary cuboids is invalid in an actual room at low frequencies.

As mentioned, only the absolute value of the uncalibrated sound pressure has been used due to the absence of phase information for the complex signal. The absolute sound pressure at the resonance frequencies recorded across the two horizontal directions marked with solid lines in Figure 4.4-4.6 has been identified as the first axial modes in the respective direction. Most of these patterns have taken on the shape of an interference pattern between an incident plane wave and a reflected one, as seen in Figure 4.7-4.14.

Furthermore, it seems that the reflected wave has been victim to a phase shift in some cases, e.g., mode (0,2,0) in room B as seen in Figure 4.11. The phase shift indicates that the impedance at that surface has a reactive part. In Table A.3, the (0,2,0) mode in room B has a more significant imaginary part in the reflection factor compared to the other modes in room B, which also would be a result of the phase shift. Moreover, for most modes with two or more nodes, it can be seen that the different minima within the same standing wave pattern have different values. These variations indicate a resistive part in the impedance as well, as described in [9, p. 74]. On the other hand, as already mentioned, there might be energy leakage from nearby modes which have not been sufficiently dampened, contributing to the spatial variations in amplitude. However, any wave pattern with an apparent symmetry between the two walls and no sudden jumps in amplitude has been deemed to sufficiently reassemble a standing wave pattern.

The least-squares-fit model has been implemented to compensate for both the resolution caused by the microphone spacing and deviations from a standing wave pattern. With few exceptions like Figure 4.10 and 4.11, the model have matched the measured values well. However, in several cases, the minima of the model end up much closer to zero than the measured values. This deviance causes the standing wave ratio to grow substantially and will cause major differences in the calculated wall characteristics, although the standing wave pattern matches the measured one well. The relationships between the modeled standing wave, which fit the measured data well, and the wall properties need to be studied further, as discussed further in section 5.5.

5.3 Modal reverberation time method

Measuring the reverberation time of individual modes has been based on somewhat the same principle as the standing wave method described. A room mode has been attempted isolated by suppressing other room modes in its spectral vicinity. The advantage of using the modal reverberation time is that it is less prone to other modes not being dampened enough, as long as it is possible to measure a sufficient amount of exponential decay of the particular mode. The disadvantage of using this method is that only the mean absorption coefficient of the two opposing walls is obtainable. That means if two walls facing each other have unknown impedances different from each other, it will not be possible to determine how much each wall contributes to the absorption of the room mode.

The first step in having this method work was to dampen modes nearby the

mode wanted to be measured. As can be seen in room A in Figure 4.15, the targeted modes are several decibels above anything else in their immediate spectral vicinity. Compared to the method described in [21], degenerate modes are not a problem when using transducer placement to reduce the closest modes, as the degenerate mode can be suppressed. In contrast to the two other rooms, the (1,0,0) and (0,1,0) modes in room B are the only ones that stand out in Figure 4.16. The (2,0,0) mode can seem to have extensive bandwidth and weak amplitude, but the (3,0,0) mode cannot be distinguished from the other modes. Both rooms B and C have a much shorter reverberation time at low frequencies than room A, which may have caused more dampened modes. Room B and C are also much smaller in volume than room A, causing the first axial modes to lay much higher in frequency. The same standing wave pattern becomes forced to exist over a shorter room dimension, causing the pressure nodes to become narrower. The narrower pressure node makes potential errors caused by transducer positioning to become more probable. Thus, more careful microphone and loudspeaker positioning may better the results.

A few criteria have to be met to evaluate the quality of the different decay curves as good enough. The first one is that the room mode must be distinguishable from the rest of the frequency response, which is true for all but the (3,0,0) mode in room B. The second criterion is that the decay must be exponential, which will be assumed true if the degree of non-linearity, ξ, is less than 10‰, as explained in section 2.8. This leaves out mode (0,1,0) in room A, mode (3,0,0) in room B, and mode (0,2,0) in room C. The last criterion is that the product of the filter bandwidth and the modal reverberation time must be greater than four and is true for all but the (2,0,0) mode in room B. However, a too narrow filter bandwidth may only extend the reverberation time, and it can still be assumed that the reverberation time is short for this particular mode. The room modes which have not been mentioned out meet all three criteria.

5.4 Absorption coefficients

In Figure 4.21, the absorption coefficient values of good quality have been enlarged, leaving measured values of poor quality easy to filter out visually. None of the values calculated through the least-squares fit model was enlarged due to the significant deviation between these values and the absorption coefficient estimated through the standing wave measurement or through the modal reverberation time measurements. Then, only considering the enlarged markers, the absorption coefficient calculated from the different methods can seem to follow a trend.

When comparing the estimated absorption coefficients of the gypsum walls, shown in Figure 4.21, with the values estimated through the wall impedance model for a 25 mm thick gypsum wall, in Figure 4.24 and 4.23, it is apparent that the first-mentioned values are way higher. Although the simplified wall impedance model carries considerable uncertainty, the difference between it and the estimations based on measurements gives reason to believe that the values estim-

ated through the standing wave ratios and modal reverberation times might be falsely high.

When comparing the absorption coefficient estimated from the standing wave method and modal reverberation time method for the glass panels to the estimations in Figure 4.24-4.26, it seems that the values from the wall impedance method are significantly lower again. The increased absorption at the different plates' structural fundamental frequency for the wall impedance model is also worth noting. These might be contributing to locally increased absorption when the frequency of the room mode coincides with the plates' fundamental frequencies. For example, it seems that for room C that the (1,0,0) mode is victim to two plates' fundamental frequency. Nevertheless, the estimated absorption coefficient from the mode, as seen in Equation 2.3 is similar to other modes close in frequency, making it hard to say whether or not the fundamental plate frequencies impact.

Unfortunately, attempts at finding literature on measurements of low-frequency wall properties have gone unsuccessful. As mentioned in the introduction, the Microflown sensor only works above 300 Hz, measurements in an impedance tube [5][6] would require a very long tube, and the reverberation room method would require an impractically large facility. Furthermore, a lab measurement would not account for the structural modes in the plate as the test sample would be smaller than an in-situ situation, and how the boundaries of the plate interact with connected structures would not be accounted for either. Thus it becomes hard to conclude whether the measurement techniques work or not, as there is no reference for comparing them. Moreover, as the estimations based on the two different measurement techniques are of similar magnitudes but are still so different from the estimation from the wall impedance predictions, it is difficult to say anything about how realistic the estimated absorption coefficients are.

5.5 Further work

As mentioned, in-situ measurements of acoustic wall properties at low frequencies are unexplored, and the validity of this report's results is unknown. On the other hand, it should provide good motivation for further exploration of this topic. Thus, a suggestion for further research of in-situ measurement of low-frequency wall properties will be suggested.

First of all, the measurement methods need validation. A suggestion to determine how well the measurement methods work is to build a smaller scale model room. By decreasing the room dimensions, the first room modes will be moved higher up in the frequency range and can thus be compared to data obtained by known methods, like the impedance tube method, two-microphone method, or a pu-sensor like the one mentioned from Microflown. A challenge that must be overcome using a much smaller room is that the pressure nodes at the first room modes will be narrower due to the smaller room dimensions. Thus accurate transducer placement becomes even more difficult. Nevertheless, if the method works

in such a small room, it is believable that it will work in a regular room.

A second suggestion is to use a measurement system that includes the time delay between the sound source and sound receiver and the non-normalized signal amplitude in the recorded impulse response. This way, the complex values of the discrete Fourier transform of the signal will be correct and correlated. Thus, instead of trying to make a least-squares-fit model for the Equation 2.19, a pseudo-inverse can be used to obtain the complex amplitudes, **A** and **B** of the two planes waves in Equation 2.18. Furthermore, the correct complex transfer function will allow attempts at obtaining the acoustic wall properties through the transfer function method for impedance tubes [6].

A more comprehensive way of determining the wall properties is to use experimental modal analysis like in, e.g., [22]. The modal parameters like resonance frequencies, damping ratios, and mode shapes are estimated from the measured frequency response recorded at many evenly spaced locations throughout the cavity. There is also possible to implement a finite element model, like in [23], using the measured boundary conditions and see if the standing wave pattern predicted by the finite element model looks anything like the measured one.

Chapter 6

Conclusion

The absorption coefficients of the walls in three different meeting rooms have been attempted estimated through two different measurement methods. Both methods focus on the center frequencies of the different axial modes within the rooms by approaching plane wave propagation through transducer positioning at the respective frequencies. Some values for the absorption coefficients have been obtained, but it is not easy to know whether the values are reliable or not.

The first method used the uncalibrated sound pressure measured in many points between two parallel walls to examine the standing wave pattern at the axial modes. Although this type of measurement is more tedious due to the number of measurements needed, it can also return much more information. Not only does it return the complex reflection factor for normal incidence of the wall, which can be converted to both the wall impedance and absorption coefficient. It also works when the opposing walls are of different constructions. One significant challenge with this method is the low spatial resolution of the measured standing waves' minima. A least-squares-fit model was used to counter this, but although the model fitted the measured standing wave pattern, the absorption coefficient estimated from the measured pattern was most similar to those estimated through the modal reverberation time method or the standing wave method.

The second method uses the modal reverberation time. The modal reverberation time was obtained by placing the transducers in different pressure nodes and applying bandpass filters to isolate the modes. In most cases, it is assumed that the modal reverberation time has been successfully measured, as the mode stands out in the frequency response and the decay rate is exponential. Thus, the modal reverberation time is converted into the mean absorption coefficient of the walls interacting with the plane wave. That means that the method only is helpful if the opposing walls are of the same construction or if one of them can be assumed rigid.

The absorption coefficients estimated through the measurement methods seem to have similar trends as a function of frequency, even though they seem to carry some uncertainty. The absorption coefficient estimated through the measurements is also higher than expected compared to a simplified estimation of the wall im-

pedance. This difference reinforces the need for further investigation of the measurement methods. It has been suggested to repeat the experiment with a scale model, such that the first room modes have a higher frequency. Other possibilities are experimental mode analysis or estimations based on a finite element model. This way, known and tested methods can determine the absorption coefficient, and the validity of these measurement methods can be established.

Bibliography

[1] J. H. Rindel, 'Searching the musical rehearsal room,' *BNAM*, May 2021.

[2] P. M. Morse and R. H. Bolt, 'Sound waves in rooms,' *Rev. Mod. Phys.*, vol. 16, pp. 69–150, 2 Apr. 1944. DOI: 10.1103/RevModPhys.16.69.

[3] S. R. Bistafa and J. W. Morrissey, 'Numerical solutions of the acoustic eigenvalue equation in the rectangular room with arbitrary (uniform) wall impedances,' *Journal of Sound and Vibration*, vol. 263, no. 1, pp. 205–218, 2003, ISSN: 0022-460X. DOI: https://doi.org/10.1016/S0022-460X(02)01123-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022460X02011239.

[4] Microflown Technologies, *In-situ absorption testing*. [Online]. Available: https://www.microflown.com/products/acoustic-material-testing/in-situ-absorption (visited on 14/04/2021).

[5] ISO-10534-1:1996(E), 'Acoustics - Determination of sound absorption coefficient and impedance in impedance tubes - Part 1: Method using standing wave ratio,' International Organization for Standardization, Geneva, CH, Standard, 1996.

[6] ISO-10534-2:1998(E), 'Acoustics - Determination of sound absorption coefficient and impedance in impedance tubes - Part 2: Transfer-function method,' International Organization for Standardization, Geneva, CH, Standard, 1998.

[7] ISO-354:2003(E), 'Acoustics - Measurement of sound absorption in a reverberation room,' International Organization for Standardization, Geneva, CH, Standard, 2003.

[8] S. Birkedal Nielsen and A. Celestinos, 'Low frequency sound field enhancement system for rectangular rooms using multiple low frequency loudspeakers,' *Journal of the Audio Engineering Society*, May 2006.

[9] H. Kuttruff, *Room acoustics*, eng, 4th ed. London: Spon Press, 2000, ISBN: 0419245804.

[10] M. Long, *Architectural Acoustics*, Second Edition. Academic Press, 2014, pp. 345–382, ISBN: 978-0-12-398258-2. DOI: https://doi.org/10.1016/B978-0-12-398258-2.00009-X.

[11] J. H. Rindel, *Sound Insulation in Buildings*. Nov. 2017, ISBN: 9781498700412. DOI: 10.1201/9781351228206.

[12] ISO-12354-1:2017(E), 'Building acoustics - Estimation of acoustic performance of buildings from the performance of elements - Part 1: Airborne sound insulation between rooms,' International Organization for Standardization, Geneva, CH, Standard, 2017.

[13] T. E. Vigran, *Building Acoustics*. 2008, ISBN: 9780429185847. DOI: https://doi.org/10.1201/9781482266016.

[14] A. Fluegel, *Calculation of Poisson's Ratio of Glasses*. [Online]. Available: http://www.glassproperties.com/poisson_ratio/ (visited on 01/06/2021).

[15] P. Sjösten, U. P. Svensson, B. Kolbrek and K. B. Evensen, 'The effect of helmholtz resonators on the acoustic room response,' *BNAM*, 2016.

[16] Mathworks. (). 'Optimizing nonlinear functions,' [Online]. Available: https://www.mathworks.com/help/matlab/math/optimizing-nonlinear-functions.html#bsgpq6p-11 (visited on 15/04/2021).

[17] F. Jacobsen and J. Rindel, 'Time reversed decay measurements,' *Journal of Sound and Vibration*, vol. 117, no. 1, pp. 187–190, 1987, ISSN: 0022-460X. DOI: https://doi.org/10.1016/0022-460X(87)90444-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022460X87904445.

[18] ISO-3382-1:2009(E), 'Acoustics - Measurement of room acoustic parameters - Part 1: Performance spaces,' International Organization for Standardization, Geneva, CH, Standard, 2009.

[19] ISO-3382-2:2008(E), 'Acoustics - Measurement of room acoustic parameters - Part 2: Reverberation time in ordinary rooms,' International Organization for Standardization, Geneva, CH, Standard, 2000.

[20] J. H. Rindel, 'A note on modal reverberation times in rectangular rooms,' *Acta Acustica united with Acustica*, vol. 102, pp. 600–603, Jan. 2016. DOI: 10.3813/AAA.918977.

[21] R. Magalotti and D. Ponteggia, 'Use of wavelet transform for the computation of modal decay times in rooms,' *Journal of the Audio Engineering Society*, Oct. 2019.

[22] G. Accardo, B. Peeters, F. Bianciardi, K. Janssens, M. El-kafafy, B. Brandolisio and M. Martarelli, 'Experimental acoustic modal analysis of an automotive cabin,' *Sound & Vibration*, 2015.

[23] M. Aretz, 'Specification of realistic boundary conditions for the fe simulation of low frequency sound fields in recording studios,' *Acta Acustica united with Acustica*, vol. 95, pp. 874–882, Sep. 2009. DOI: 10.3813/AAA.918219.

Appendix A

Supplementary Material

A.1 Equipment list

Table A.1: Equipment list for all measurements

Description	Manufacturer	Model	Serial
Omnidirectional loudspeaker	Odeon	Prototype	Unknown
Microphone power supply	Norsonic	336	Unknown
Audio Interface	Behringer	UCA202	Unknown
Microphone	Norsonic	1220	19998
Pre-amplifier	Norsonic	1201	14324
Microphone cable 20m	Norsonic	P1408A	140-B
Stand	Unknown	Unknown	Unknown
Stand	Manfrotto	MS0490A	Unknown
RCA-BCN cable 1m	Unknown	Unknown	Unknown
Computer running Odeon	Dell	Unknown	MCO-3861
Odeon combined, v.16	Odeon	Unknown	Unknown
Laser measuring tool	Bosch	DLE30	#1
Measurement tape	Unknown	Unknown	Unknown
String	Unknown	Unknown	Unknown
Glass Measurement Gauge	Merlin Lazer	Unknown	Unknown

A.2 Modal reverberation time measurement positions

Table A.2: Coordinate of source and receiver during modal reverberation time measurement.

		Source			Receiver		
Room	**Mode**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
A	100	1.5 m	3.07 m	1.7 m	5.2 m	1.55 m	0.85
	200	3.2 m	3.07 m	1.7 m	5.2 m	1.55 m	0.85 m
	300	1.5 m	3.07 m	1.7 m	0.75 m	1.55 m	0.85 m
	010	3.2 m	5.05 m	1.7 m	1.5 m	5.05 m	0.85 m
	020	3.2 m	3.07 m	1.7 m	1.5 m	5.05 m	0.85 m
	030	3.2 m	5.05 m	1.7 m	1.5 m	0.78 m	0.85 m
B	100	0.95 m	1.1 m	1.7 m	2.9 m	0.55 m	0.85 m
	200	1.75 m	1.1 m	1.7 m	2.9 m	0.55 m	0.85 m
	300	0.95 m	1.1 m	1.7 m	0.95 m	0.55 m	0.85 m
	010	1.75 m	0.55 m	1.7 m	2.9 m	0.55 m	0.85 m
C	100	0.75 m	1.15 m	1.7 m	2.55 m	0.6 m	0.85 m
	200	1.5 m	1.15 m	1.7 m	2.55 m	0.6 m	0.85 m
	300	0.75 m	1.15 m	1.7 m	1.5 m	0.6 m	0.85 m
	010	1.5 m	0.6 m	1.7 m	0.75 m	1.15 m	0.85 m
	020	1.5 m	1.15 m	1.7 m	0.75 m	1.15 m	0.85 m

A.3 Reverberation times from Odeon

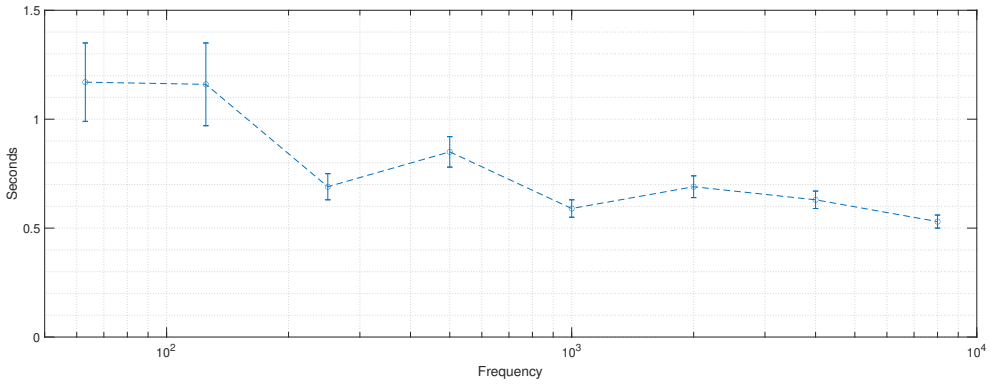


Figure A.1: Reverberation time T20 and standard deviation in room A.

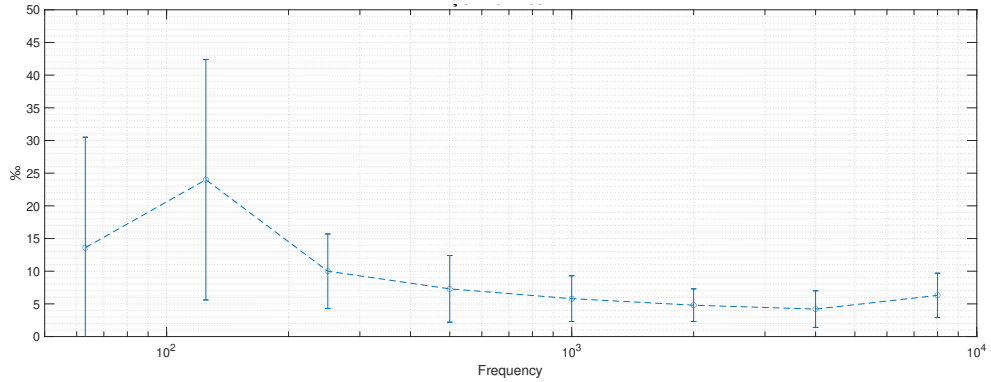


Figure A.2: Degree of non-linear decay, ξ, for T20 in room A.

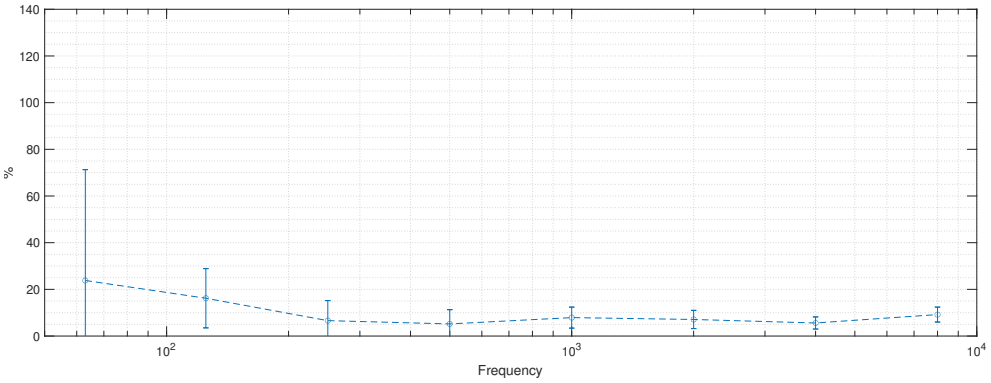


Figure A.3: Curvature of the decay rate in room A.

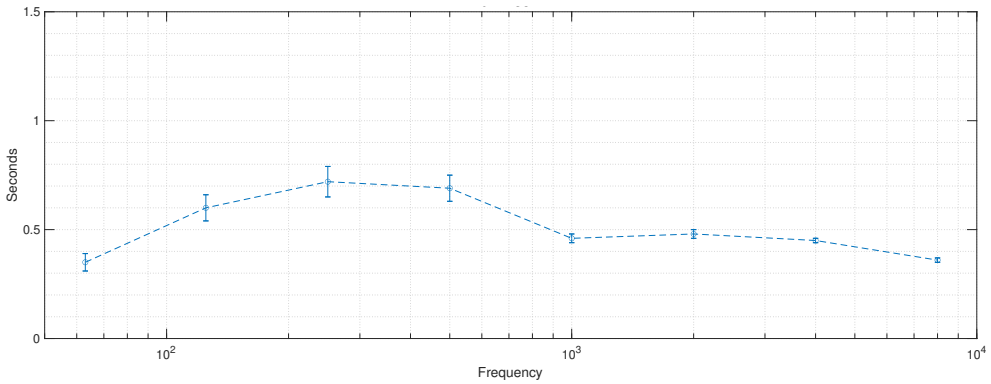


Figure A.4: Reverberation time T20 and standard deviation in room B.

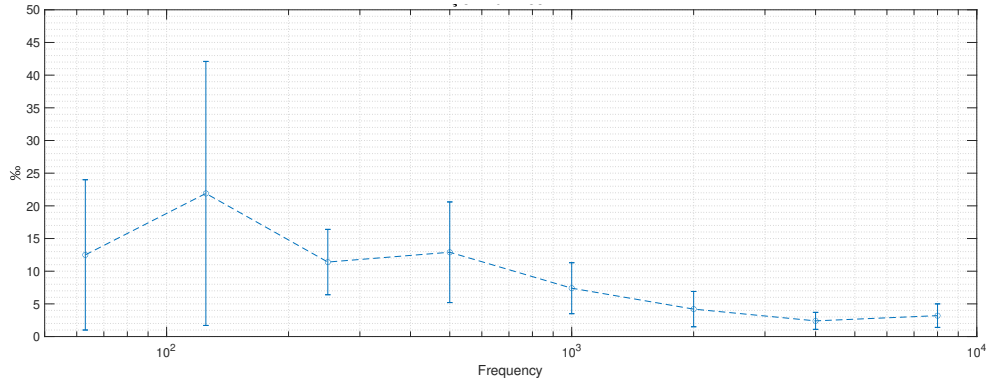


Figure A.5: Degree of non-linear decay, ξ, for T20 in room B.

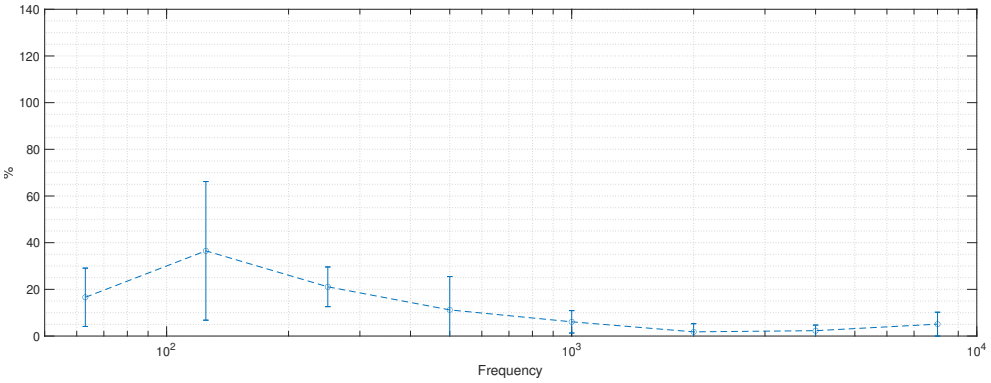


Figure A.6: Curvature of the decay rate in room B.

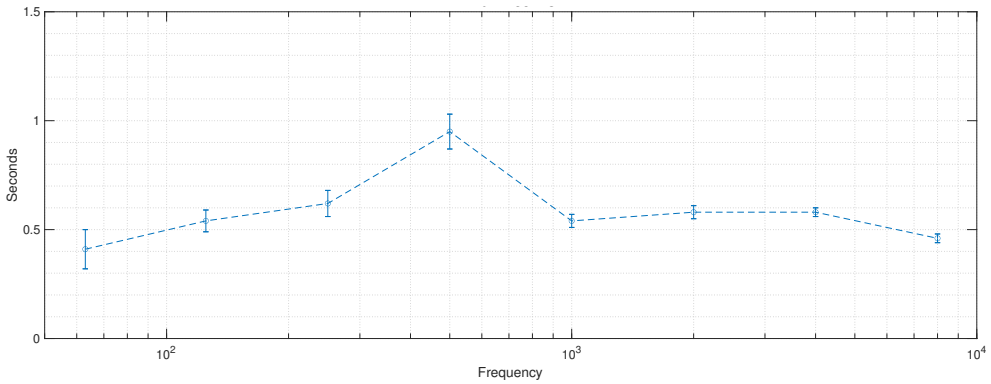


Figure A.7: Reverberation time T20 and standard deviation in room C.

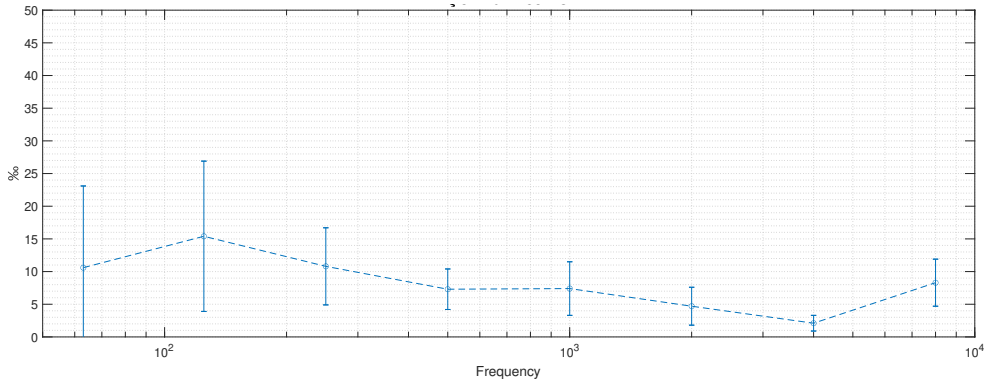


Figure A.8: Degree of non-linear decay, ξ, for T20 in room C.

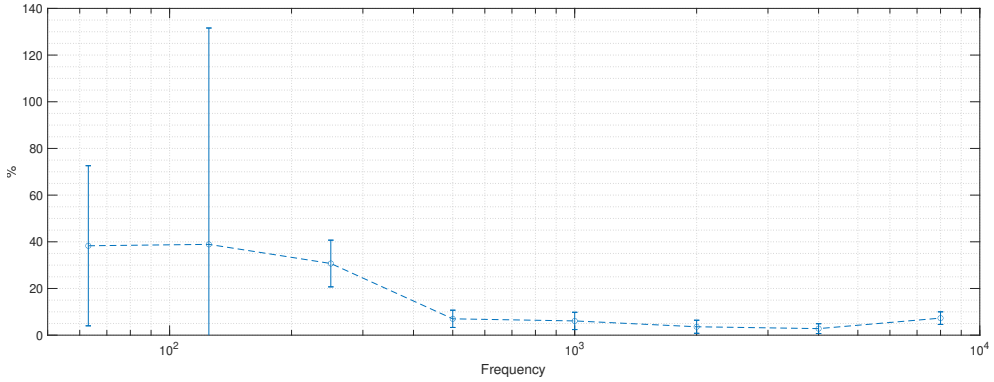


Figure A.9: Curvature of the decay rate in room C.

A.4 Standing wave ratio results

Table A.3: Reflection coefficient, characteristic impedance and absorption coefficient calculated through standing wave method.

Room	Mode	Frequency	Standing wave ratio	Reflection factor	Characteristic impedance	Absorption coefficient
A	100	28.1 Hz	15.62	0.864+i0.166	4.913+i7.207	0.226
	200	54.8 Hz	14.45	0.867-i0.084	9.754-i6.747	0.242
	300	82.1 Hz	4.26	0.607+i0.126	3.615+i1.474	0.616
	010	29.6 Hz	8.83	0.663+i0.441	1.19+i2.864	0.366
	020	57.0 Hz	17.58	0.867+i0.210	3.312+6.814	0.204
	030	85.1 Hz	4.47	0.601+i0.203	2.978+i2.027	0.598
B	100	52.8 Hz	7.30	0.729+i0.211	3.597+i3.580	0.424
	200	102.0 Hz	10.73	0.802+i0.213	3.689+i5.033	0.312
	300	150.0 Hz	10.48	0.815+i0.158	5.223+i5.191	0.318
	010	82.4 Hz	10.34	0.804+i0.177	4.627+i5.088	0.322
	020	160.0 Hz	10.92	0.762+i0.335	1.815+i3.961	0.307
C	100	61.7 Hz	9.02	0.768+i0.223	3.410+i4.302	0.359
	200	118.0 Hz	19.02	0.844+i0.313	1.548+i5.111	0.190
	010	78.7 Hz	10.20	0.803+i0.172	4.752+i5.036	0.325
	020	154.0 Hz	3.48	0.554-i0.002	3.481-i0.021	0.693

Table A.4: Reflection coefficient, characteristic impedance and absorption coefficient calculated through standing wave method.

Room	Mode	Frequency	Standing wave ratio	Reflection factor	Characteristic impedance	Absorption coefficient
A	100	28.1 Hz	391.23	0.965+i0.242	0.170+i8.100	0.010
	200	54.8 Hz	9.43	0.808-i0.008	9.404-i0.460	0.347
	300	82.1 Hz	3.18	0.519+i0.050	3.116+i0.431	0.728
	010	29.6 Hz	14.01	0.741+i0.450	0.921+i3.334	0.249
	020	57.0 Hz	14.32	0.856+i0.154	5.474+i6.913	0.244
	030	85.1 Hz	5.09	0.629+i0.236	2.834+i2.441	0.549
B	100	52.8 Hz	295.75	0.873+i0.474	0.056+i3.932	0.013
	200	102.0 Hz	217.32	0.970+i0.201	0.440+i9.713	0.018
	300	150.0 Hz	6.60	0.717+i0.169	4.213+i3.116	0.457
	010	82.4 Hz	283.79	0.987+i0.108	1.194+i18.340	0.014
	020	160.0 Hz	4.88	0.622+i0.219	2.964+i2.297	0.565
C	100	61.7 Hz	16.13	0.840+i0.274	2.181+i5.437	0.220
	200	118.0 Hz	281.22	0.937+i0.329	0.126+i5.854	0.014
	010	78.7 Hz	160.12	0.921+i0.356	0.186+i5.357	0.0247
	020	154.0 Hz	2.70	0.439-i0.137	2.362-i0.823	0.789

Appendix B

Matlab code

Code listing B.1: WavToPf.m

```
function Pf= WavToPf(file,Nfft,RT,delay)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OUTPUTS:
% Pf: Uncalibrated sound pressure in the frequency domain
% INPUTS:
% file: path to impulseresponse of a room in .wav format
% Nfft: Size of the DFT
% maxf: maximum frequency concidered
% RT: Reverberation time
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[y,fs] = audioread(file); % Importing impulse response signal as a vector, ...
    and its sampling frequency
scale = odeonAttuFactor(file);
y = y./scale;
if nargin>5
    y = cat(1,zeros(delay,1),y);
end
fvec(:,1) = fs*(0:(Nfft/2-1))/Nfft; % Makingg frequency vector
Y = fft(y(1:RT*fs*0.8),Nfft,1); % DFT of the impulse response, truncation ...
    is based on the reverberation time
y_LSP = audioread('0-180_averageTimeDomain.wav'); % importing impulse ...
    response of the loudspeaker measurement
Y_LSP = fft(y_LSP(1:7000),Nfft,1); % DFT of a reference measurement of the ...
    loudspeaker, truncated to remove noise
Pf = Y./Y_LSP;
Pf = Pf(1:Nfft/2);
end
```

Code listing B.2: odeonAttuFactor.m

```
function AttuFactor = odeonAttuFactor(filename)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This function returns the attenuation factor, that odeon has multiplied
% the impulse response by to make its maximum value equal to 1. By dividing
% by the obtained attenuation factor, the original amplitude of the
% impulseresponse is restored.
```

```
7  fid = fopen(filename, 'r','l'); %Opens file
8  data = fread(fid,40,'uint'); %Imports data
9  ddata = dec2hex(data(12)); %Converts the data to hexadecimal
10 AttuFactor = (typecast(uint32(hex2dec(ddata)),'single')); %Converts the ...
       hexadecimal to the attenuation factor
11 fclose(fid); %closes the file
12 end
```

Code listing B.3: modalsum_rigidroom2.m

```
1  function [pressure,nmodes,resfreqs] = ...
       modalsum_rigidroom2(source,receiver,freqvec,roomsize,T60,rhoair,cair,safetyfactor)
2  % This function calculates the sound pressure in a room, caused by a point ...
       source,
3  % as calculated by a modal sum. A harmonic source is assumed with the volume
4  % acceleration 1 m/s^2.
5  %
6  % Input parameters:
7  %   source          A vector, [xs,ys,zs], giving the position of the ...
       source in meters.
8  %   receiver        A vector, [xr,yr,zr], giving the position of the ...
       receiver in meters.
9  %   freqvec         A vector, size [1,nfreqs], with the frequency values ...
       of the source signal in Hz.
10 %   roomsize        A vector, [lx,ly,lz], giving the room dimensions in meters.
11 %   T60             The reverberation time of the room. A constant value ...
       for all frequencies
12 %                   is assumed.
13 %   rhoair          The density of air.
14 %   cair            The speed of sound in air.
15 %   safetyfactor    (optional) The ratio between the lowest missed mode ...
       resonance frequency and
16 %                   the highest frequency of interest. Default value: 1.2.
17 %
18 % Output parameters:
19 %   pressure        A vector, size [1,nfreqs], with the complex amplitudes ...
       of the sound pressure
20 %                   for the nfreqs frequency values of the frequency.
21 %   nmodes          The number of modes that were used.
22 %   resfreqs        The resonance frequencies of the used modes.
23 %
24 % Peter Svensson, 15 Sept. 2018  [peter.svensson@ntnu.no]
25 %
26 % [pressure,nmodes,resfreqs] = ...
       modalsum_rigidroom2(source,receiver,freqvec,roomsize,T60,rhoair,cair,safetyfactor);
27
28 % Diary:
29 % 020405 First version
30 % 020408 Added damping
31 % 5 Feb. 2016 Added the modeamplitudes as output parameter
32 % 15 Sep. 2018 Added comments, and made the safetyfactor an optional
33 % parameter
34
35 if nargin < 8
36    safetyfactor = 1.2;
37 end
38
39 % Enforce the freqvec to be a horizontal vector since that fits with the matrix
```

```
% we want to construct further below (of size [nmodes,nfreqs])
freqvec = freqvec(:).';

nfreqs = length(freqvec);
maxfreq = max(freqvec);

%---------------------------------------------------------------
% Construct a matrix [nmodes,3] of all mode numbers, combinations
% of integers [0,1,2,...].
%
% To get absolutely all modes with a resonance frequency which is
% lower than a certain value (the frequency times the safetyfactor),
% we must find the maximum integer in the three directions.

% Find the highest possible mode number in each direction.

nxmax = ceil(2*roomsize(1)*maxfreq*safetyfactor/cair)+1;
nymax = ceil(2*roomsize(2)*maxfreq*safetyfactor/cair)+1;
nzmax = ceil(2*roomsize(3)*maxfreq*safetyfactor/cair)+1;

nmodes = nxmax*nymax*nzmax;

% Construct first repeated vectors for the x and y directions
% The function meshgrid gives matrices that we convert to vertical vectors

nxymodes = nxmax*nymax;
[vecx,vecy] = meshgrid([0:nxmax-1].',[0:nymax-1].');
vecx = reshape(vecx,nxymodes,1);
vecy = reshape(vecy,nxymodes,1);

% Now we repeat the matrix [vecx vecy] (with two columns) nzmax times in
% the vertical direction...

qvec = repmat([vecx vecy],nzmax,1);

% ...and add a third column where the z-mode number is added.
% We first make a horizontal vector of 0,1,2,....,
% then we repeat that row nxymax times,
% and finally reshape it into a vertical vector which has the same length
% as the qvec matrix, so we append the qvec matrix with thirdcolumn

thirdcolumn = [0:nzmax-1];
thirdcolumn = repmat(thirdcolumn,nxymodes,1);
thirdcolumn = reshape(thirdcolumn,nmodes,1);

qvec = [qvec,thirdcolumn];

% qvec should now be a matrix like this (if nymax = 3)
% qvec = [0 0 0;0 1 0;0 2 0;0 3 0;1 1 0;1 2 0;1 3 0; etc]

% Now we have all modes with a resonance frequency below
% max(freqvec)*safetyfactor but we want to keep only those that
% have a resonance frequency below max(freqvec)*safetyfactor.

% So, we calculate the resonance frequency of all the modes.
% It can be done in a single line

resfreqs = cair/2*sqrt( sum(((qvec./roomsize(ones(nmodes,1),:)).^2).') ).';

ivec = find(resfreqs <= maxfreq*safetyfactor);
```

```
100 qvec = qvec(ivec,:);
101 resfreqs = resfreqs(ivec);
102 clear ivec
103
104 % Update the number of modes that have been kept
105 nmodes = size(qvec,1);
106
107 %---------------------------------------------------------------------
108 % Calculate the mode functions' values for the source and receiver
109
110 modeamp = cos(qvec(:,1)*pi*source(1)/roomsize(1)) ...
        .*cos(qvec(:,2)*pi*source(2)/roomsize(2)) ...
        .*cos(qvec(:,3)*pi*source(3)/roomsize(3));
111
112 % The mode amplitude also has a factor which depends on how many of the three
113 % integers are zero.
114
115 modescalefactor = sign(qvec)+1;
116 modescalefactor = prod(modescalefactor.').';
117 modeamp = modeamp.*modescalefactor;
118
119 % Now modeamp is a vector of size [modeamp,1]
120
121 %---------------------------------------------------------------------
122 % We expand the modeamp vector into a matrix of size [nmodes,nfreqs] by
123 % including the frequency-dependent part of the mode amplitude.
124
125 delta = 3*log(10)/T60;
126
127 if nfreqs > 1
128     onesvec2 = ones(1,nfreqs);
129
130     modeamp = modeamp(:,onesvec2)./(resfreqs(:,onesvec2).^2 - ...
            freqvec(ones(nmodes,1),:).^2 + ...
            1i*2*delta*resfreqs(:,onesvec2)/2/pi );
131     modefunctionval = cos(qvec(:,1)*pi*receiver(1)/roomsize(1)) ...
            .*cos(qvec(:,2)*pi*receiver(2)/roomsize(2)) ...
            .*cos(qvec(:,3)*pi*receiver(3)/roomsize(3));
132     modeamp = modeamp.*modefunctionval(:,onesvec2);
133
134     if nmodes > 1
135         pressure = sum( modeamp );
136     else
137         pressure = modeamp;
138     end
139 else
140
141     modeamp = modeamp./(resfreqs.^2 - freqvec.^2 + 1i*2*delta*resfreqs/2/pi);
142     modeamp = modeamp.*cos(qvec(:,1)*pi*receiver(1)/roomsize(1)) ...
            .*cos(qvec(:,2)*pi*receiver(2)/roomsize(2)) ...
            .*cos(qvec(:,3)*pi*receiver(3)/roomsize(3));
143     if nmodes > 1
144         pressure = sum( modeamp );
145     else
146         pressure = modeamp;
147     end
148
149 end
150
151 % Since we summed over all the modes above, pressure is a vector of size
```

```
% [1,nfreqs]. We multiply with the scale factor which is usually written
% outside the modal sum expression.

pressure = 1i*2*pi*freqvec.*pressure*rhoair*cair^2/prod(roomsize)/4/pi^2;
```

Code listing B.4: Shoeboxfreq.m

```
function [resfreqs,FSI] = Shoeboxfreq(roomsize,freqvec,cair);

% THE FOLLOWING PART IS COPYIED FROM modalsum_rigidroom2.m by Peter
% Svennson:
safetyfactor = 1.2;
freqvec = freqvec(:).';
maxfreq = max(freqvec);
nxMax = ceil(2*roomsize(1)*maxfreq*safetyfactor/cair)+1;
nyMax = ceil(2*roomsize(2)*maxfreq*safetyfactor/cair)+1;
nzMax = ceil(2*roomsize(3)*maxfreq*safetyfactor/cair)+1;
nmodes = nxMax*nyMax*nzMax;
nfreqs = length(freqvec);
nxymodes = nxMax*nyMax;
[vecx,vecy] = meshgrid([0:nxMax-1].',[0:nyMax-1].');
vecx = reshape(vecx,nxymodes,1);
vecy = reshape(vecy,nxymodes,1);
qvec = repmat([vecx vecy],nzMax,1);
thirdcolumn = [0:nzMax-1];
thirdcolumn = repmat(thirdcolumn,nxymodes,1);
thirdcolumn = reshape(thirdcolumn,nmodes,1);
qvec = [qvec,thirdcolumn];
resfreqs(:,1) = cair/2*sqrt( ...
    sum(((qvec./roomsize(ones(nmodes,1),:)).^2).') ).';
resfreqs = [resfreqs,qvec];

% THE FOLLOWING PART IS ADDED TO CALCULATE THE FREQUENCY SPACING INDEX FROM
% THE NATURAL FREQUENCIES CALCULATED ABOVE.
[~,idf] = sort(resfreqs,1);
resfreqs25 = resfreqs(idf(2:26),:);

avg_freq_space = (resfreqs25(end,1)-resfreqs25(1,1))/(size(resfreqs25,1)-1);
freq_space = resfreqs25(2:end,1) - resfreqs25(1:end-1,1);
FSI = (1/(resfreqs25(end,1)-resfreqs25(1,1))) * ...
    sum((freq_space.^2)/avg_freq_space);
```

Code listing B.5: RoomPrepA.m

```
close all
clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script plots predicted frequency response for a rigid room,
% predicted natural frequencies and the measured frequency from corner to
% corner in room A.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Parameters:
Lx = 6.3;
Ly = 6.15;
Lz = 3.4;
T = 1;
```

```
c = 339;
Nfft = 2^18;
FS = 44100;
maxfreq = 200;
minfreq = 20;
%% Calculatinng eigenfrequencies and predicted FR
dims = [Lx Ly Lz];
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft;
ivf = find(fvec < maxfreq);
eigenfreqs = Shoeboxfreq(dims,fvec(ivf),c);
[~,idf] = sort(eigenfreqs,1);
eigenfreqs = eigenfreqs(idf,:);
p = modalsum_rigidroom2([0 0 0],dims,fvec(ivf),dims,T,1.19,c);
L = 20*log10(abs(p));
L_room2 = WavToSPL('Pilot/Impulse response ...
     file.ImpRespFile2.wav',Nfft,FS,maxfreq,T);

%% Plotting
figure()
plot(fvec(ivf),L(ivf)-40,'-b','Linewidth',1)
hold on
%plot(fvec(ivf),L_room(ivf),'-k','Linewidth',1)
plot(fvec(ivf),L_room2(ivf),'-k','Linewidth',2)
toffset = 2;
for i = 2:25
    xline(eigenfreqs(i,1),'--k','LineWidth',0.5)
    txtin = ...
         ['\leftarrow(',num2str(eigenfreqs(i,2)),',',num2str(eigenfreqs(i,3)),',',num2str(eigenfreqs(i,4))
    text(eigenfreqs(i,1),-18+abs(toffset),txtin,'VerticalAlignment','bottom','Color','r','FontSize',12,'Fo
    H=findobj(gca,'Type','text');
    set(H,'Rotation',0)
    if rem(toffset,10)==0
        toffset = 2;
    else
        toffset = (toffset+2);
    end
end
hold off
title('Predicted and measured corner to corner frequency response in Room A')
xlabel('Frequency [Hz]')
ylabel('[dB]')
legend({'Predicted frequency response','Measured corner to corner ...
     response'},'Location','best')
set(gca, 'fontsize', 15)
xlim([20 120])
ylim([-20 25])
set(gca, 'fontsize', 16)
```

Code listing B.6: RoomPrepB.m

```
close all
clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script plots predicted frequency response for a rigid room,
% predicted natural frequencies and the measured frequency from corner to
% corner in room B.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Parameters:
```

```
Lx = 3.5;
Ly = 2.2;
Lz = 3.4;
T = 0.5;
c = 339;
Nfft = 2^18;
FS = 44100;
maxfreq = 200;
minfreq = 20;
%% Calculatinng eigenfrequencies and predicted FR
dims = [Lx Ly Lz];
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft;
ivf = find(fvec < maxfreq);
eigenfreqs = Shoeboxfreq(dims,fvec(ivf),c);
[~,idf] = sort(eigenfreqs,1);
eigenfreqs = eigenfreqs(idf,:);
p = modalsum_rigidroom2([0 0 0],dims,fvec(ivf),dims,T,1.19,c);
L = 20*log10(abs(p));
L_room2 = WavToSPL('B524/TEsts/CornerCorner.wav',Nfft,FS,maxfreq,T);

%% Plotting
figure()
plot(fvec(ivf),L(ivf)-50,'-b','Linewidth',1)
hold on
plot(fvec(ivf),L_room2(ivf),'-k','Linewidth',2)
toffset = 2;
for i = 2:25
    xline(eigenfreqs(i,1),'--k','LineWidth',0.5)
    txtin = ...
        ['\leftarrow(',num2str(eigenfreqs(i,2)),',',num2str(eigenfreqs(i,3)),',',num2str(eigenfreqs(i,4))
    text(eigenfreqs(i,1),-15+abs(toffset),txtin,'VerticalAlignment','bottom','Color','r','FontSize',12,'Fo
    H=findobj(gca,'Type','text');
    set(H,'Rotation',0)
    if rem(toffset,10)==0
        toffset = 2;
    else
        toffset = (toffset+2);
    end
end
hold off
title('Predicted and measured corner to corner frequency response in Room B')
xlabel('Frequency [Hz]')
ylabel('[dB]')
legend({'Predicted frequency response','Measured corner to corner ...
    response'},'Location','best')
set(gca, 'fontsize', 15)
xlim([20 200])
ylim([-15 20])
```

Code listing B.7: RoomPrepC.m

```
close all
clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script plots predicted frequency response for a rigid room,
% predicted natural frequencies and the measured frequency from corner to
% corner in room C.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%% Parameters:
Lx = 3.05;
Ly = 2.2;
Lz = 3.4;
T = 0.5;
c = 339;
Nfft = 2^18;
FS = 44100;
maxfreq = 200;
minfreq = 20;
%% Calculatinng eigenfrequencies and predicted FR
dims = [Lx Ly Lz];
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft;
ivf = find(fvec < maxfreq);
eigenfreqs = Shoeboxfreq(dims,fvec(ivf),c);
[~,idf] = sort(eigenfreqs,1);
eigenfreqs = eigenfreqs(idf,:);
p = modalsum_rigidroom2([0 0 0],dims,fvec(ivf),dims,T,1.19,c);
L = 20*log10(abs(p));
L_room2 = WavToSPL('B553/Test/CornerCorner.wav',Nfft,FS,maxfreq,T);

%% Plotting
figure()
plot(fvec(ivf),L(ivf)-56,'-b','Linewidth',1)
hold on
plot(fvec(ivf),L_room2(ivf),'-k','Linewidth',2)
toffset = 2;
for i = 2:25
    xline(eigenfreqs(i,1),'--k','LineWidth',0.5)
    txtin = ...
        ['\leftarrow(',num2str(eigenfreqs(i,2)),',',num2str(eigenfreqs(i,3)),',',num2str(eigenfreqs(i,4))
    text(eigenfreqs(i,1),-18+abs(toffset),txtin,'VerticalAlignment','bottom','Color','r','FontSize',12,'Fo
    H=findobj(gca,'Type','text');
    set(H,'Rotation',0)
    if rem(toffset,10)==0
        toffset = 2;
    else
        toffset = (toffset+2);
    end
end
hold off
title('Predicted and measured corner to corner frequency response in Room C')
xlabel('Frequency [Hz]')
ylabel('[dB]')
legend({'Predicted frequency response','Measured corner to corner ...
    response'},'Location','best')
set(gca, 'fontsize', 15)
xlim([20 180])
ylim([-20 15])
```

Code listing B.8: Room.m

```
classdef Room < handle
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % This class calculates imports the impulse responses of the respective
    % room and calculates a least-squares-fit model to it. It requires the
    % files to lay in a folder path equal to "\<Room name>\LineX\" or
    % "\<Room name>\LineY\" with filenames "Impulse response
```

```
7       % file.ImpRespFile0", where the number 0 would be the impulse response
8       % measured closest to the wall in question.
9       %
10      % Functions:
11      %   inputvals:      Prompts the user to input the room variables in command
12      %                   window
13      %   fill:           Imports data from impulse responses and performs all
14      %                   calculations.
15      %   plotpressure:   Plots the standing wave pattern by pressure.
16      %   plotpressuresquare :    Plots the standing wave pattern by square
17      %                           pressure.
18      %   plotlevel:      Plots the standing wave pattern by level
19      %
20      %
21      % Author : Henrik K. R. Berg
22      % E-mail : hkrb94@gmail.com
23      properties
24          Roomname
25          dim
26          T
27          Xvec
28          Yvec
29          ivX
30          ivY
31          Nfft
32          Rp_x
33          Rp_y
34          Zs_x
35          Zs_y
36          ivf
37          modenumber
38          modefreq
39          Xmodefreq
40          Ymodefreq
41      end
42      properties (Hidden)
43          fvec
44          fs %Schroeder frequency
45          px %pressure along xaxis
46          py %pressure along yaxis
47          px_model %pressure least-squares-fit model along xaxis
48          py_model %pressure least-squares-fit model along yaxis
49          c %sound of speed
50          Xhd %Vector for rooms x dimension in high res
51          Yhd %Vector for rooms x dimension in high res
52      end
53      methods
54          function obj = inputvals(obj)
55              obj.Roomname = convertCharsToStrings(input('Input room name:   ...
                    ','s'));
56              L_x = input('Input length L_x:   ');
57              L_y = input('Input length L_y:   ');
58              L_z = input('Input length L_z:   ');
59              obj.dim = [L_x,L_y,L_z];
60              obj.T = input('Input reverberation time:   ');
61              obj.Nfft = input('Input number of FFT points:   ');
62              Xvec(:,1) = input(['Declare X-vector from 0 to ...
                    ',num2str(-L_x,3),':   ']);
63              obj.Xvec = Xvec;
64              Yvec(:,1) = input(['Declare Y-vector from 0 to ...
```

```
                ',num2str(-L_y,3),':   ']);
65          obj.Yvec = Yvec;
66      return;
67      end
68      function obj = fill(obj)
69          obj.fvec(1:obj.Nfft/2,1) = 44100*(0:(obj.Nfft/2-1))/obj.Nfft;
70          obj.c = 340;
71          obj.Xhd(:,1) = obj.Xvec(1):-0.01:obj.Xvec(end);
72          obj.Yhd(:,1) = obj.Yvec(1):-0.01:obj.Yvec(end);
73          %% Schroeder frequency
74          if numel(obj.T) == 1 && numel(obj.dim) == 3;
75              obj.fs = 2000*sqrt(obj.T/prod(obj.dim));
76              obj.ivf = find(obj.fvec <obj.fs);
77          else
78              disp("Missing reverberation time T and/or room dimensions ...
                    dim [Lx, Ly, Lz].")
79          end
80          %% Importing wav files
81          if isstring(obj.Roomname)==1
82              Xdir = dir(strcat(obj.Roomname,"/LineX/*.wav"));
83              obj.px = zeros(numel(Xdir),obj.Nfft/2);
84              Ydir = dir(strcat(obj.Roomname,'/LineY/*.wav'));
85              obj.py = zeros(numel(Ydir),obj.Nfft/2);
86              for i =0:1:numel(Xdir)-1
87                  obj.px(i+1,:) = ...
                        WavToPf(strcat(obj.Roomname,['/LineX/Impulse ...
                        response ...
                        file.ImpRespFile',num2str(i),'.wav']),obj.Nfft,obj.T);
88              end
89              for i = 0:1:numel(Ydir)-1
90                  obj.py(i+1,:) = ...
                        WavToPf(strcat(obj.Roomname,['/LineY/Impulse ...
                        response ...
                        file.ImpRespFile',num2str(i),'.wav']),obj.Nfft,obj.T);
91              end
92          else
93              disp("Missing Roomname.")
94          end
95          %% Calculating a mimimum error model for pressure in X and Y ...
                directions
96          [obj.px_model,obj.py_model,obj.Zs_x,obj.Zs_y,obj.Rp_x,obj.Rp_y] ...
                = Fminsearch(obj.px,obj.py,obj.Xvec,obj.Yvec, ...
                obj.Xhd,obj.Yhd,obj.ivX,obj.ivY,obj.fvec,obj.fs);
97          %% Natural frequencies for rigid room
98          eigenfreqs = Shoeboxfreq(obj.dim,obj.fvec(obj.ivf),obj.c);
99          [~,idf] = sort(eigenfreqs,1);
100         eigenfreqs = eigenfreqs(idf,:);
101         obj.modenumber = ...
                [num2str(eigenfreqs(:,2),1),num2str(eigenfreqs(:,3),1), ...
                num2str(eigenfreqs(:,4),1)];
102         obj.modefreq = eigenfreqs(:,1);
103         for i = 1:1:4
104         obj.Xmodefreq(i,1) = ...
                obj.modefreq(find(eigenfreqs(:,2)==i,1,'first'));
105         obj.Ymodefreq(i,1) = ...
                obj.modefreq(find(eigenfreqs(:,3)==i,1,'first'));
106         end
107     end
108     %% Plotting functions
109     function obj = plotpressure(obj)
```

```
110             figure(1)
111             sgtitle(['Frequency pattern in frequency and 2 dimensions from ...
                    room ',obj.Roomname,'Left click to pick frequency. ...
                    Right-click to stop.'])
112             subplot(2,2,1)
113             plot(obj.fvec(obj.ivf),20*log10(abs(obj.px(:,obj.ivf))));
114             for i = 1:numel(obj.Xmodefreq)
115                 xline(obj.Xmodefreq(i,1))
116             end
117             title('Pressure in  the X-direction in by frequency'), ...
                    xlabel('Frequency [Hz]'),ylabel('[dB]');
118             xlim([20 obj.fs])
119             set(gca, 'fontsize', 18)
120             subplot(2,2,3)
121             plot(obj.fvec(obj.ivf),20*log10(abs(obj.py(:,obj.ivf))));
122             for i = 1:numel(obj.Ymodefreq)
123                 xline(obj.Ymodefreq(i,1))
124             end
125             xlim([20 obj.fs])
126             title('Pressure in  the Y-direction in by frequency'), ...
                    xlabel('Frequency [Hz]'),ylabel('[dB]');
127             set(gca, 'fontsize', 18)
128             btn = 0;
129             while btn ~= 3
130                 [pos,~,btn] = ginput(1);
131                 freq = interp1(obj.fvec,obj.fvec,pos,'nearest');
132                 freqid = find(obj.fvec == freq);
133                 figure(1)
134                 subplot(2,2,2)
135                 plot(obj.Xvec,(abs(obj.px(:,freqid))), ...
                        obj.Xhd,obj.px_model(:,freqid))
136                 xlim([obj.Xvec(end) obj.Xvec(1)])
137                 title(['Pressure in X-direction at f = ',num2str(freq,3),' ...
                        Hz']),xlabel('Meters'),ylabel('Amplitude'), ...
                        legend({'p_{x-dir,measured}','p_{x-dir,estimated}'}, ...
                        'Location','best')
138                 subplot(2,2,4)
139                 plot(obj.Yvec,(abs(obj.py(:,freqid))), ...
                        obj.Yhd,obj.py_model(:,freqid))
140                 xlim([obj.Yvec(end) obj.Yvec(1)])
141                 title(['Pressure in Y-direction at f = ',num2str(freq,3),' ...
                        Hz']),xlabel('Meters'),ylabel('Amplitude'), ...
                        legend({'p_{x-dir,measured}','p_{x-dir,estimated}'}, ...
                        'Location','best')
142             end
143
144         end
145         function obj = plotpressuresquare(obj)
146             figure(1)
147             sgtitle(['Frequency pattern in frequency and 2 dimensions from ...
                    room ',obj.Roomname,'Left click to pick frequency. ...
                    Right-click to stop.'])
148             subplot(2,2,1)
149             plot(obj.fvec(obj.ivf),20*log10(abs(obj.px(:,obj.ivf))));
150             for i = 1:numel(obj.Xmodefreq)
151                 xline(obj.Xmodefreq(i,1))
152             end
153             title('Pressure in  the X-direction in by ...
                    frequency'),xlabel('Frequency [Hz]'),ylabel('[dB]');
154             xlim([20 obj.fs])
```

```
155             set(gca, 'fontsize', 18)
156             subplot(2,2,3)
157             plot(obj.fvec(obj.ivf),20*log10(abs(obj.py(:,obj.ivf))));
158             for i = 1:numel(obj.Ymodefreq)
159                 xline(obj.Ymodefreq(i,1))
160             end
161             xlim([20 obj.fs])
162             title('Pressure in  the Y-direction in by ...
                    frequency'),xlabel('Frequency [Hz]'),ylabel('[dB]');
163             set(gca, 'fontsize', 18)
164             btn = 0;
165             while btn ~= 3
166                 [pos,~,btn] = ginput(1);
167                 freq = interp1(obj.fvec,obj.fvec,pos,'nearest');
168                 freqid = find(obj.fvec == freq);
169                 figure(1)
170                 subplot(2,2,2)
171                 plot(obj.Xvec,(abs(obj.px(:,freqid))).^2, ...
                        obj.Xhd,obj.px_model(:,freqid).^2)
172                 xlim([obj.Xvec(end) obj.Xvec(1)])
173                 title(['Pressure in X-direction at f = ',num2str(freq,3),' ...
                        Hz']),xlabel('Meters'),ylabel('Amplitude'), ...
                        legend({'p_{x-dir,measured}^2','p_{x-dir,estimated}^2'}, ...
                        'Location','best')
174                 subplot(2,2,4)
175                 plot(obj.Yvec,(abs(obj.py(:,freqid))).^2, ...
                        obj.Yhd,obj.py_model(:,freqid).^2)
176                 xlim([obj.Yvec(end) obj.Yvec(1)])
177                 title(['Pressure in Y-direction at f = ',num2str(freq,3),' ...
                        Hz']),xlabel('Meters'),ylabel('Amplitude'), ...
                        legend({'p_{x-dir,measured}^2','p_{x-dir,estimated}^2'}, ...
                        'Location','best')
178             end
179         end
180         function obj = plotlevel(obj)
181             figure(1)
182             sgtitle(['Frequency pattern in frequency and 2 dimensions from ...
                    room ',obj.Roomname,'Left click to pick frequency. ...
                    Right-click to stop.'])
183             subplot(2,2,1)
184             plot(obj.fvec(obj.ivf),20*log10(abs(obj.px(:,obj.ivf))));
185             for i = 1:numel(obj.Xmodefreq)
186                 xline(obj.Xmodefreq(i,1))
187             end
188             title('Pressure in  the X-direction in by ...
                    frequency'),xlabel('Frequency [Hz]'),ylabel('[dB]');
189             xlim([20 obj.fs])
190             set(gca, 'fontsize', 18)
191             subplot(2,2,3)
192             plot(obj.fvec(obj.ivf),20*log10(abs(obj.py(:,obj.ivf))));
193             for i = 1:numel(obj.Ymodefreq)
194                 xline(obj.Ymodefreq(i,1))
195             end
196             xlim([20 obj.fs])
197             title('Pressure in  the Y-direction in by ...
                    frequency'),xlabel('Frequency [Hz]'),ylabel('[dB]');
198             set(gca, 'fontsize', 18)
199             btn = 0;
200             while btn ~= 3
201                 [pos,~,btn] = ginput(1);
```

```
202                 freq = interp1(obj.fvec,obj.fvec,pos,'nearest');
203                 freqid = find(obj.fvec == freq);
204                 figure(1)
205                 subplot(2,2,2)
206                 plot(obj.Xvec,20*log10(abs(obj.px(:,freqid))), ...
                        obj.Xhd,20*log10(obj.px_model(:,freqid)))
207                 xlim([obj.Xvec(end) obj.Xvec(1)])
208                 title(['Sound level in X-direction at f = ...
                        ',num2str(freq,3),' ...
                        Hz']),xlabel('Meters'),ylabel('Amplitude'), ...
                        legend({'L_{p,x-dir,measured}','L_{p,x-dir,estimated}'}, ...
                        'Location','best')
209                 set(gca, 'fontsize', 18)
210                 subplot(2,2,4)
211                 plot(obj.Yvec,20*log10(abs(obj.py(:,freqid))), ...
                        obj.Yhd,20*log10(obj.py_model(:,freqid)))
212                 xlim([obj.Yvec(end) obj.Yvec(1)])
213                 title(['Sound level in Y-direction at f = ...
                        ',num2str(freq,3),' ...
                        Hz']),xlabel('Meters'),ylabel('Amplitude'), ...
                        legend({'L_{p,x-dir,measured}','L_{p,x-dir,estimated}'}, ...
                        'Location','best')
214                 set(gca, 'fontsize', 18)
215             end
216         end
217         function obj = plotimpedance(obj)
218             figure(2)
219             sgtitle('Specific impedance of walls')
220             subplot(1,2,1)
221             plot(obj.fvec(obj.ivf),real(obj.Zs_x),imag(obj.Zs_x))
222             title('At X = 0'),xlabel('Frequency ...
                    [Hz]'),ylabel('Z_s'),legend({'Real','Imag'},'Location','best')
223             set(gca, 'fontsize', 18)
224             ylim([-20 20])
225             subplot(1,2,2)
226             plot(obj.fvec(obj.ivf),real(obj.Zs_y),imag(obj.Zs_y))
227             title('At Y = 0'),xlabel('Frequency ...
                    [Hz]'),ylabel('Z_s'),legend({'Real','Imag'},'Location','best')
228             set(gca, 'fontsize', 18)
229             ylim([-20 20])
230         end
231         function obj = plotreflection(obj)
232             figure(2)
233             sgtitle('Complex reflection factor of walls')
234             subplot(1,2,1)
235             plot(obj.fvec(obj.ivf),real(obj.Rp_x),obj.fvec(obj.ivf),imag(obj.Rp_x))
236             title('At X = 0'),xlabel('Frequency ...
                    [Hz]'),ylabel('R_p'),legend({'Real','Imag'},'Location','best')
237             set(gca, 'fontsize', 18)
238             ylim([0 2])
239             subplot(1,2,2)
240             plot(obj.fvec(obj.ivf),real(obj.Rp_y),obj.fvec(obj.ivf),imag(obj.Rp_y))
241             title('At Y = 0'),xlabel('Frequency ...
                    [Hz]'),ylabel('R_p'),legend({'Real','Imag'},'Location','best')
242             set(gca, 'fontsize', 18)
243             ylim([0 2])
244         end
245         function obj = plotabsreflection(obj)
246             figure(2)
247             sgtitle('Complex reflection factor of walls')
```

```
248             subplot(1,2,1)
249             plot(obj.fvec(obj.ivf),abs(obj.Rp_x))
250             title('At X = 0'),xlabel('Frequency [Hz]'),ylabel('|R_p|'),
251             set(gca, 'fontsize', 18)
252             ylim([0 2])
253             subplot(1,2,2)
254             plot(obj.fvec(obj.ivf),abs(obj.Rp_y))
255             title('At Y = 0'),xlabel('Frequency [Hz]'),ylabel('|R_p|'),
256             set(gca, 'fontsize', 18)
257             ylim([0 2])
258         end
259     end
260 end
```

Code listing B.9: Fminsearch.m

```
1  function [px_model,py_model,Zs_x,Zs_y,Rp_x,Rp_y] = ...
       Fminsearch(px,py,Xvec,Yvec,Xhd,Yhd,ivX,ivY,fvec,fschroeder)
2  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
3  % This function calculate the least-squares-fit model of the standing wave
4  % pattern measured from many positions along two dimensions of a room.
5  % INPUTS:
6  %           px          :   2D array of frequency responses along x-axis
7  %           py          :   2D array of frequency responses along y-axis
8  %           Xvec        :   Position vector of all microphone positions ...
       along x-axis
9  %           Yvec        :   Position vector of all microphone positions ...
       along y-axis
10 %           Xhd         :   Output spatial resolution of the model for x-axis
11 %           Yhd         :   Output spatial resolution of the model for y-axis
12 %           ivX         :   Index vector of mic positions to use for x-axis
13 %           ivY         :   Index vector of mic positions to use for y-axis
14 %           fvec        :   frequency vector
15 %           fschroeder  :   max frequency that will be modelled.
16 % OUTPUTS:
17 %           px_model    :   Modelled standing wave pattern along x-axis
18 %           py_model    :   Modelled standing wave pattern along y-axis
19 %           Zs_x        :   Calculated specific acoustic impedance in x=0
20 %           Zs_y        :   Calculated specific acoustic impedance in y=0
21 %           Rp_x        :   Calculated reflection factor in x=0
22 %           Rp_y        :   Calculated reflection factor in y=0
23 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
24 c=342;
25 ivf = find(fvec < fschroeder);
26 kvec = fvec*2*pi/c;
27 for i= 1:numel(fvec(ivf))
28 ii = 1;
29 options = optimset('fminsearch');
30 options.MaxFunEvals = 10000000;
31 options.MaxIter = 100000;
32
33 % Inital guesses for y-axis measurements
34 maxmin_ini = max(abs(py(ivY,i)).^2)/min(abs(py(ivY,i)).^2);
35 absRp_ini = (sqrt(maxmin_ini)-1)/(sqrt(maxmin_ini)+1);
36 absAsquare_ini = max(abs(py(ivY,i)).^2)/(1+absRp_ini)^2;
37 phi_ini = 2*kvec(i)*Yvec(find(py(ivY,i).^2 == min(py(ivY,i).^2)))-pi;
38 val_guess = double([absAsquare_ini absRp_ini phi_ini]);
39 % Finding best fitting coefficients
```

```
40  val = fminsearch(@(vals) ...
        findpsquare(vals,py(ivY,i),kvec(i),Yvec(ivY)),val_guess,options);
41  % Results for y-axis at frequency index i
42  absPysquare(i,:) = val(1)*(1+val(2)^2+2*val(2)*cos(2*kvec(i)*Yhd(:)+val(3)));
43  absRpy(i,1) = val(2);
44  Rpyangle(i,1) = val(3);
45  % Inital guesses for x-axis measurements
46  maxmin_ini = max(abs(px(ivX,i)).^2)/min(abs(px(ivX,i)).^2);
47  absRp_ini = (sqrt(maxmin_ini)-1)/(sqrt(maxmin_ini)+1);
48  absAsquare_ini = max(abs(px(ivX,i)).^2)/(1+absRp_ini)^2;
49  phi_ini = 2*kvec(i)*Xvec(find(px(ivX,i).^2 == min(px(ivX,i).^2)))-pi;
50  val_guess = double([absAsquare_ini absRp_ini phi_ini]);
51  % Finding best fitting coefficients
52  val = fminsearch(@(vals) ...
        findpsquare(vals,px(ivX,i),kvec(i),Xvec(ivX)),val_guess,options);
53  % Results for x-axis at frequency index i
54  absPxsquare(i,:) = val(1)*(1+val(2)^2+2*val(2)*cos(2*kvec(i)*Xhd(:)+val(3)));
55  absRpx(i,1) = val(2);
56  Rpxangle(i,1) = val(3);
57  end
58  Rp_x = (absRpx.*cos(Rpxangle) + 1i*absRpx.*sin(Rpxangle)).';
59  Rp_y = (absRpy.*cos(Rpyangle) + 1i*absRpy.*sin(Rpyangle)).';
60  Zs_x = ((1 + Rp_x) ./ (1-Rp_x));
61  Zs_y = ((1 + Rp_y) ./ (1-Rp_y));
62  px_model = sqrt(absPxsquare).';
63  py_model = sqrt(absPysquare).';
64  end
```

Code listing B.10: findpsquare.m

```
1  function [epsilonsquare] = findpsquare(val,py,k,x)
2  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
3  % Function calculating epsilon squared between model and measurement.
4  % This is the function that is minimized in order to find the
5  % least-squares-fit model of the standing wave
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7      absAsquare = val(1);
8      absRp = val(2);
9      phi = val(3);
10     psquare_model = (absAsquare*(1+absRp^2+2*absRp*cos(2*k*x+phi)));
11     psquare_measurement = abs(py(:)).^2;
12     epsilonsquare = sum((psquare_model-psquare_measurement).^2);
13 end
```

Code listing B.11: initrooms.m

```
1  close all
2  clear
3  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4  % This script initiate the room objects in the same way they were used in
5  % the thesis.
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7  RoomA = Room;
8  RoomA.Roomname = "C510";
9  RoomA.dim = [6.3,6.15,3.4];
10 RoomA.T = 1.2;
```

```
RoomA.Xvec(:,1) = [0:-0.1:-6.3];
RoomA.Yvec(:,1) = [0:-0.1:-6.0 -6.15];
RoomA.Nfft = 2^18;
RoomA.ivX = [6:30 32:57];
RoomA.ivY = [6:56];

RoomB = Room;
RoomB.Roomname = "B524";
RoomB.dim = [3.5,2.2,3.4];
RoomB.T = 0.5;
RoomB.Xvec(:,1) = [0:-0.05:-3.5];
RoomB.Yvec(:,1) = [0:-0.05:-2.2];
RoomB.Nfft = 2^18;
RoomB.ivX = [11:61];
RoomB.ivY = [6:40];

RoomC = Room;
RoomC.Roomname = "B553";
RoomC.dim = [3.05,2.3,3.4];
RoomC.T = 0.5;
RoomC.Xvec(:,1) = [0:-0.05:-3.05];
RoomC.Yvec(:,1) = [0:-0.05:-2.3];
RoomC.Nfft = 2^18;
RoomC.ivX = [6:57];
RoomC.ivY = [6:41];

RoomA.fill;
RoomB.fill;
RoomC.fill;

save("RoomA.mat","RoomA");
save("RoomB.mat","RoomB");
save("RoomC.mat","RoomC");
```

Code listing B.12: PLOTS.m

```
close all
clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script loads the room objects and plots their frequency responses
% and the standing wave pattern for the first 2-3 axial modes in both the
% X- and the Y-directions.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
load('RoomA.mat')
load('RoomB.mat')
load('RoomC.mat')
AresX = [28.1 54.8 82.1]; % Discorvered resonance in X-direction in room A
AresY = [29.6 57 84.8]; % Discorvered resonance in Y-direction in room A
BresX = [52.8 102 150]; % Discorvered resonance in X-direction in room B
BresY = [82.4 160]; % Discorvered resonance in Y-direction in room B
CresX = [61.7 118]; % Discorvered resonance in X-direction in room C
CresY = [78.7 154]; % Discorvered resonance in Y-direction in room C
ivAX = [1:30 32:41 43:64]; % Indexvector leaving bad IRs in room A's ...
    X-direction out of the plots.
%% Frequency response
figure()
sgtitle('Room A')
subplot(2,1,1)
```

```
plot(RoomA.fvec(RoomA.ivf),20*log10(abs(RoomA.px(ivAX,RoomA.ivf))))
for i=1:3
xline(RoomA.Xmodefreq(i),'--k')
xline(AresX(i),'-k')
text(AresX(i),-50,['\leftarrow ...
    ',num2str(i),'00'],'FontSize',11,'FontWeight','bold')
end
title('Frequency response along X-axis')
xlabel('Frequency [Hz]')
ylabel('Amplitude [dB]')
xlim([20 RoomA.fs])
set(gca, 'fontsize', 14)
subplot(2,1,2)
plot(RoomA.fvec(RoomA.ivf),20*log10(abs(RoomA.py(:,RoomA.ivf))))
for i=1:3
xline(RoomA.Ymodefreq(i),'--k')
xline(AresY(i),'-k')
text(AresY(i),-50,['\leftarrow ...
    0',num2str(i),'0'],'FontSize',11,'FontWeight','bold')
end
title('Frequency response along Y-axis')
xlabel('Frequency [Hz]')
ylabel('Amplitude [dB]')
set(gca, 'fontsize', 14)
xlim([20 RoomA.fs])

figure()
sgtitle('Room B')
subplot(2,1,1)
plot(RoomB.fvec(RoomB.ivf),20*log10(abs(RoomB.px(:,RoomB.ivf))))
for i=1:3
xline(RoomB.Xmodefreq(i),'--k')
xline(BresX(i),'-k')
text(BresX(i),-50,['\leftarrow ...
    ',num2str(i),'00'],'FontSize',11,'FontWeight','bold')
end
title('Frequency response along X-axis')
xlabel('Frequency [Hz]')
ylabel('Amplitude [dB]')
xlim([20 RoomB.fs])
set(gca, 'fontsize', 14)
subplot(2,1,2)
plot(RoomB.fvec(RoomB.ivf),20*log10(abs(RoomB.py(:,RoomB.ivf))))
for i=1:2
xline(RoomB.Ymodefreq(i),'--k')
xline(BresY(i),'-k')
text(BresY(i),-50,['\leftarrow ...
    0',num2str(i),'0'],'FontSize',11,'FontWeight','bold')
end
title('Frequency response along Y-axis')
xlabel('Frequency [Hz]')
ylabel('Amplitude [dB]')
set(gca, 'fontsize', 14)
xlim([20 RoomB.fs])

figure()
sgtitle('Room C')
subplot(2,1,1)
plot(RoomC.fvec(RoomC.ivf),20*log10(abs(RoomC.px(:,RoomC.ivf))))
for i=1:2
```

```
78  xline(RoomC.Xmodefreq(i),'--k')
79  xline(CresX(i),'-k')
80  text(CresX(i),-50,['\leftarrow ...
        ',num2str(i),'00'],'FontSize',11,'FontWeight','bold')
81  end
82  title('Frequency response along X-axis')
83  xlabel('Frequency [Hz]')
84  ylabel('Amplitude [dB]')
85  xlim([20 RoomC.fs])
86  set(gca, 'fontsize', 14)
87  subplot(2,1,2)
88  plot(RoomC.fvec(RoomC.ivf),20*log10(abs(RoomC.py(:,RoomC.ivf))))
89  for i=1:2
90  xline(RoomC.Ymodefreq(i),'--k')
91  xline(CresY(i),'-k')
92  text(CresY(i),-50,['\leftarrow ...
        0',num2str(i),'0'],'FontSize',11,'FontWeight','bold')
93  end
94  title('Frequency response along Y-axis')
95  xlabel('Frequency [Hz]')
96  ylabel('Amplitude [dB]')
97  set(gca, 'fontsize', 14)
98  xlim([20 RoomC.fs])
99
100 %% Room A Spatial response
101 figure()
102 set(gcf,'position',[0,0,1800,600])
103 sgtitle('Room A, mode 100 & 010')
104 subplot(1,2,1)
105 plot(RoomA.Xvec(ivAX),abs(RoomA.px(ivAX,168)),'-k', ...
        RoomA.Xvec(RoomA.ivX),abs(RoomA.px(RoomA.ivX,168)),'or', ...
        RoomA.Xhd,RoomA.px_model(:,168),'-b')
106 title('Sound pressure along X-axis at f = 28.1 Hz')
107 xlabel('Distance from reflecting surface [m]')
108 ylabel('|p(\omega)|')
109 legend('Measured points','Points used in the model','Least-squares-fit model')
110 xlim([RoomA.Xvec(end) 0])
111 set(gca, 'fontsize', 14)
112 subplot(1,2,2)
113 plot(RoomA.Yvec,abs(RoomA.py(:,177)),'-k', ...
        RoomA.Yvec(RoomA.ivY),abs(RoomA.py(RoomA.ivY,177)),'or', ...
        RoomA.Yhd,RoomA.py_model(:,177),'-b')
114 title('Sound pressure along Y-axis at f = 29.6 Hz')
115 xlabel('Distance from reflecting surface [m]')
116 ylabel('|p(\omega)|')
117 set(gca, 'fontsize', 14)
118 xlim([RoomA.Yvec(end) 0])
119
120 figure()
121 set(gcf,'position',[0,0,1800,600])
122 sgtitle('Room A, mode 200 & 020')
123 subplot(1,2,1)
124 plot(RoomA.Xvec(ivAX),abs(RoomA.px(ivAX,327)),'-k', ...
        RoomA.Xvec(RoomA.ivX),abs(RoomA.px(RoomA.ivX,327)),'or', ...
        RoomA.Xhd,RoomA.px_model(:,327),'-b')
125 title('Sound pressure along X-axis at f = 54.8 Hz')
126 xlabel('Distance from reflecting surface [m]')
127 ylabel('|p(\omega)|')
128 xlim([RoomA.Xvec(end) 0])
129 set(gca, 'fontsize', 14)
```

```
subplot(1,2,2)
plot(RoomA.Yvec,abs(RoomA.py(:,340)),'-k', ...
     RoomA.Yvec(RoomA.ivY),abs(RoomA.py(RoomA.ivY,340)),'or', ...
     RoomA.Yhd,RoomA.py_model(:,340),'-b')
title('Sound pressure along Y-axis at f = 57.0 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
set(gca, 'fontsize', 14)
xlim([RoomA.Yvec(end) 0])

figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('Room A, mode 300 & 030')
subplot(1,2,1)
plot(RoomA.Xvec(ivAX),abs(RoomA.px(ivAX,483)),'-k', ...
     RoomA.Xvec(RoomA.ivX),abs(RoomA.px(RoomA.ivX,483)),'or', ...
     RoomA.Xhd,RoomA.px_model(:,483),'-b')
title('Sound pressure along X-axis at f = 82.1 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
xlim([RoomA.Xvec(end) 0])
set(gca, 'fontsize', 14)
subplot(1,2,2)
plot(RoomA.Yvec,abs(RoomA.py(:,505)),'-k', ...
     RoomA.Yvec(RoomA.ivY),abs(RoomA.py(RoomA.ivY,505)),'or', ...
     RoomA.Yhd,RoomA.py_model(:,505),'-b')
title('Sound pressure along Y-axis at f = 84.8 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
set(gca, 'fontsize', 14)
xlim([RoomA.Yvec(end) 0])

%% Room B
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('Room B, mode 100 & 010')
subplot(1,2,1)
plot(RoomB.Xvec,abs(RoomB.px(:,315)),'-k', ...
     RoomB.Xvec(RoomB.ivX),abs(RoomB.px(RoomB.ivX,315)),'or', ...
     RoomB.Xhd,RoomB.px_model(:,315),'-b')
title('Sound pressure along X-axis at f = 52.8 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
xlim([RoomB.Xvec(end) 0])
set(gca, 'fontsize', 14)
subplot(1,2,2)
plot(RoomB.Yvec,abs(RoomB.py(:,491)),'-k', ...
     RoomB.Yvec(RoomB.ivY),abs(RoomB.py(RoomB.ivY,491)),'or', ...
     RoomB.Yhd,RoomB.py_model(:,491),'-b')
title('Sound pressure along Y-axis at f = 82.4 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
set(gca, 'fontsize', 14)
xlim([RoomB.Yvec(end) 0])

figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('Room B, mode 200 & 020')
subplot(1,2,1)
plot(RoomB.Xvec,abs(RoomB.px(:,607)),'-k', ...
```

```
        RoomB.Xvec(RoomB.ivX),abs(RoomB.px(RoomB.ivX,607)),'or', ...
        RoomB.Xhd,RoomB.px_model(:,607),'-b')
title('Sound pressure along X-axis  at f = 102.0 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
xlim([RoomB.Xvec(end) 0])
set(gca, 'fontsize', 14)
subplot(1,2,2)
plot(RoomB.Yvec,abs(RoomB.py(:,952)),'-k', ...
        RoomB.Yvec(RoomB.ivY),abs(RoomB.py(RoomB.ivY,952)),'or', ...
        RoomB.Yhd,RoomB.py_model(:,952),'-b')
title('Sound pressure along Y-axis at f = 160.0 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
set(gca, 'fontsize', 14)
xlim([RoomB.Yvec(end) 0])

figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('Room B, mode 300')
subplot(1,2,1)
plot(RoomB.Xvec,abs(RoomB.px(:,893)),'-k', ...
        RoomB.Xvec(RoomB.ivX),abs(RoomB.px(RoomB.ivX,893)),'or', ...
        RoomB.Xhd,RoomB.px_model(:,893),'-b')
title('Sound pressure along X-axis at f = 150.0 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
xlim([RoomB.Xvec(end) 0])
set(gca, 'fontsize', 14)
% subplot(1,2,2)
% plot(RoomB.Yvec,abs(RoomB.py(:,505)),'-k', ...
        RoomB.Yvec(RoomB.ivY),abs(RoomB.py(RoomB.ivY,505)),'or', ...
        RoomB.Yhd,RoomB.py_model(:,505),'--b')
% title('Sound pressure along Y-axis in X = and Z =')
% xlabel('Distance from reflecting surface]')
% ylabel('|p(\omega)|')
% set(gca, 'fontsize', 14)
% xlim([RoomB.Yvec(end) 0])

%% Room C
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('Room C, mode 100 & 010')
subplot(1,2,1)
plot(RoomC.Xvec,abs(RoomC.px(:,368)),'-k', ...
        RoomC.Xvec(RoomC.ivX),abs(RoomC.px(RoomC.ivX,368)),'or', ...
        RoomC.Xhd,RoomC.px_model(:,368),'-b')
title('Sound pressure along X-axis at f = 61.7 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
xlim([RoomC.Xvec(end) 0])
set(gca, 'fontsize', 14)
subplot(1,2,2)
plot(RoomC.Yvec,abs(RoomC.py(:,469)),'-k', ...
        RoomC.Yvec(RoomC.ivY),abs(RoomC.py(RoomC.ivY,469)),'or', ...
        RoomC.Yhd,RoomC.py_model(:,469),'--b')
title('Sound pressure along Y-axis at f = 78.7 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
set(gca, 'fontsize', 14)
```

```
xlim([RoomC.Yvec(end) 0])

figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('Room C, mode 200 & 020')
subplot(1,2,1)
plot(RoomC.Xvec,abs(RoomC.px(:,703)),'-k', ...
    RoomC.Xvec(RoomC.ivX),abs(RoomC.px(RoomC.ivX,703)),'or', ...
    RoomC.Xhd,RoomC.px_model(:,703),'-b')
title('Sound pressure along X-axis at f = 118.0 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
xlim([RoomC.Xvec(end) 0])
set(gca, 'fontsize', 14)
subplot(1,2,2)
plot(RoomC.Yvec,abs(RoomC.py(:,917)),'-k', ...
    RoomC.Yvec(RoomC.ivY),abs(RoomC.py(RoomC.ivY,917)),'or', ...
    RoomC.Yhd,RoomC.py_model(:,917),'-b')
title('Sound pressure along Y-axis at f = 154.0 Hz')
xlabel('Distance from reflecting surface [m]')
ylabel('|p(\omega)|')
set(gca, 'fontsize', 14)
xlim([RoomC.Yvec(end) 0])
```

Code listing B.13: ModalT_C510.m

```
%% Init
clear
close all

Lx = 6.3; % Length
Ly = 6.15; % Width
Lz = 3.4; % Height
T = 1.2; % Reverberation time at low frequencies
c = 340; % speed of air in the room
Nfft = 2^19; % FFT Size
FS = 44100; % Samplerate
maxfreq = 120; % maximum frequency plotted
minfreq = 20; % minimum frequency plotted
dims = [Lx Ly Lz];
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft; % Frequency vector
tvec(:,1) = (1:1*44101)/44100; % Time vector for 1 second
ivf = find(fvec < maxfreq); % frequency index
eigenfreqs = Shoeboxfreq(dims,fvec(ivf),c);
[~,idf] = sort(eigenfreqs,1);
eigenfreqs = eigenfreqs(idf,:); % sorted natural frequency for rigid walls
dirac = [1; zeros(136709,1)]; % unit pulse
AresX = [28.1 54.8 82.1]; % First axial modal frequencies observed for the ...
    X axis
AresY = [29.6 57 84.8]; % First axial modal frequencies observed for the X axis
%% The following section imports the impulse responses
%% LSP
yLSP = audioread('0-180_averageTimeDomain.wav');
YLSP = fft(yLSP(1:7000),Nfft,1);
%% 100
y100 = audioread('Modaletterklang/100.wav');
Y100 = fft(y100(1:T*100000),Nfft,1)./YLSP;
%% 010
```

```
32
33 y010 = audioread('Modaletterklang/010.wav');
34 Y010 = fft(y010(1:T*100000),Nfft,1)./YLSP;
35 %% 110
36 y110_0 = audioread('Modaletterklang/110(0.0).wav');
37 y110_1 = audioread('Modaletterklang/110(Ly.Lx).wav');
38
39 y110 = y110_0-y110_1;
40 Y110 = fft(y110(1:T*100000),Nfft,1)./YLSP;
41 %% 001
42 y001 = audioread('Modaletterklang/001.wav');
43 Y001 = fft(y001(1:T*100000),Nfft,1)./YLSP;
44 %% 200
45 y200 = audioread('Modaletterklang/200.wav');
46 Y200 = fft(y200(1:T*100000),Nfft,1)./YLSP;
47 %% 020
48 y020 = audioread('Modaletterklang/020.wav');
49 Y020 = fft(y020(1:T*100000),Nfft,1)./YLSP;
50 %% 101
51 y101 = audioread('Modaletterklang/101.wav');
52 Y101 = fft(y101(1:T*100000),Nfft,1)./YLSP;
53 %% 011
54 y011 = audioread('Modaletterklang/011.wav');
55 Y011 = fft(y011(1:T*100000),Nfft,1)./YLSP;
56 %% 210
57 y210_0 = audioread('Modaletterklang/210(0).wav');
58 y210_1 = audioread('Modaletterklang/210(Ly).wav');
59
60 y210 = y210_0 - y210_1;
61 Y210 = fft(y210(1:T*100000),Nfft,1)./YLSP;
62 %% 120
63 y120_0 = audioread('Modaletterklang/120(0).wav');
64 y120_1 = audioread('Modaletterklang/120(Lx).wav');
65
66 y120 = y120_0 + y120_1;
67 Y120 = fft(y120(1:T*100000),Nfft,1)./YLSP;
68 %% 111
69 y111 = audioread('Modaletterklang/111.wav');
70 Y111 = fft(y111(1:T*50000),Nfft,1)./YLSP;
71 %% 201
72 y201 = audioread('Modaletterklang/201.wav');
73 Y201 = fft(y201(1:T*100000),Nfft,1)./YLSP;
74 %% 021
75 y021 = audioread('Modaletterklang/021.wav');
76 Y021 = fft(y021(1:T*100000),Nfft,1)./YLSP;
77 %% 220
78 y220 = audioread('Modaletterklang/220.wav');
79 Y220 = fft(y220(1:T*100000),Nfft,1)./YLSP;
80 %% 211
81 y211 = audioread('Modaletterklang/211.wav');
82 Y211 = fft(y211(1:T*100000),Nfft,1)./YLSP;
83 %% 121
84 y121 = audioread('Modaletterklang/121.wav');
85 Y121 = fft(y121(1:T*100000),Nfft,1)./YLSP;
86 %% 300
87 y300 = audioread('Modaletterklang/300.wav');
88 Y300 = fft(y300(1:T*100000),Nfft,1)./YLSP;
89 %% 030
90 y030 = audioread('Modaletterklang/030.wav');
91 Y030 = fft(y030(1:T*100000),Nfft,1)./YLSP;
```

```
%% 310
y310_0 = audioread('Modaletterklang/310(0).wav');
y310_1 = audioread('Modaletterklang/310(Ly).wav');

y310 = y310_0 + y310_1;
Y310 = fft(y310(1:T*100000),Nfft,1)./YLSP;
%% 130
y130_0 = audioread('Modaletterklang/130(0).wav');
y130_1 = audioread('Modaletterklang/130(Lx).wav');

y130 = y130_0 - y130_1;
Y130 = fft(y130(1:T*100000),Nfft,1)./YLSP;
%% 221
y221 = audioread('Modaletterklang/221.wav');
Y221 = fft(y221(1:T*100000),Nfft,1)./YLSP;
%% 301
y301 = audioread('Modaletterklang/301.wav');
Y301 = fft(y301(1:T*100000),Nfft,1)./YLSP;
%% 031
y031 = audioread('Modaletterklang/031.wav');
Y031 = fft(y031(1:T*100000),Nfft,1)./YLSP;
%% 320
y320 = audioread('Modaletterklang/320.wav');
Y320 = fft(y320(1:T*100000),Nfft,1)./YLSP;
%% 311
y311 = audioread('Modaletterklang/311.wav');
Y311 = fft(y311(1:T*100000),Nfft,1)./YLSP;
%% 002
y002 = audioread('Modaletterklang/002.wav');
Y002 = fft(y002(1:T*100000),Nfft,1)./YLSP;
%% The following sections backwards filter the IRs before they are ...
     backwards integrated. Then a least-squares line is calculated and all ...
     the parameters are calculated. Like T_modal, Xi, and alpha.
%% 100 filtering
octfilt100 = octaveFilter(30.1,'2/3 octave');
filter100 = octfilt100(dirac);
y100filt = flip(octfilt100(flip(y100)));
Y100filter = fft(filter100,Nfft);
y100filt = y100filt(1:35000);
D100 = 10*log10(flip(cumsum(flip(y100filt.^2))));
D100 = D100-max(D100);
idD100 = find(D100 <= -5 & D100 >= -25);
p100 = polyfit(tvec(idD100),D100(idD100),1);
R100 = polyval(p100,tvec);
T100 = 60/(R100(1)-R100(end));
rsq100 = (sum((R100(idD100)-mean(D100(idD100))).^2)) / ...
     (sum((D100(idD100)-mean(D100(idD100))).^2));
nonlin100 = 1000*(1-rsq100);
alphax100 = 55.3*Lx^2*28.1/(-c^2*1*T100);
alpham100 = 1-exp(alphax100);
Y100filt = fft(y100filt,Nfft,1);
%% 010 filtering
octfilt010 = octaveFilter(29.6,'2/3 octave');
filter010 = octfilt010(dirac);
y010filt = flip(octfilt010(flip(y010)));
Y010filter = fft(filter010,Nfft);
y010filt = y010filt(1:16000);
D010 = 10*log10(flip(cumsum(flip(y010filt.^2))));
D010 = D010-max(D010);
idD010 = find(D010 <= -5 & D010 >= -25);
```

```
149 p010 = polyfit(tvec(idD010),D010(idD010),1);
150 R010 = polyval(p010,tvec);
151 T010 = 60/(R010(1)-R010(end));
152 rsq010 = (sum((R010(idD010)-mean(D010(idD010))).^2)) / ...
        (sum((D010(idD010)-mean(D010(idD010))).^2));
153 nonlin010 = 1000*(1-rsq010);
154 alphax010 = 55.3*Ly^2*29.6/(-c^2*1*T010);
155 alpham010 = 1-sqrt(exp(alphax010));
156 Y010filt = fft(y010filt,Nfft,1);
157
158 %% 200 filtering
159 octfilt200 = octaveFilter(52.8,'1/3 octave');
160 filter200 = octfilt200(dirac);
161 y200filt = flip(octfilt200(flip(y200)));
162 Y200filter = fft(filter200,Nfft);
163 y200filt = y200filt(1:40000);
164 D200 = 10*log10(flip(cumsum(flip(y200filt.^2))));
165 D200 = D200-max(D200);
166 idD200 = find(D200 <= -5 & D200 >= -25);
167 p200 = polyfit(tvec(idD200),D200(idD200),1);
168 R200 = polyval(p200,tvec);
169 T200 = 60/(R200(1)-R200(end));
170 rsq200 = (sum((R200(idD200)-mean(D200(idD200))).^2)) / ...
        (sum((D200(idD200)-mean(D200(idD200))).^2));
171 nonlin200 = 1000*(1-rsq200);
172 alphax200 = 55.3*Lx^2*54.8/(-c^2*2*T200);
173 alpham200 = 1-exp(alphax200);
174 Y200filt = fft(y200filt,Nfft,1);
175 %% 020 filtering
176 octfilt020 = octaveFilter(57,'1/3 octave');
177 filter020 = octfilt020(dirac);
178 y020filt = flip(octfilt020(flip(y020)));
179 Y020filter = fft(filter020,Nfft);
180 y020filt = y020filt(1:41000);
181 D020 = 10*log10(flip(cumsum(flip(y020filt.^2))));
182 D020 = D020-max(D020);
183 idD020 = find(D020 <= -5 & D020 >= -25);
184 p020 = polyfit(tvec(idD020),D020(idD020),1);
185 R020 = polyval(p020,tvec);
186 T020 = 60/(R020(1)-R020(end));
187 rsq020 = (sum((R020(idD020)-mean(D020(idD020))).^2)) / ...
        (sum((D020(idD020)-mean(D020(idD020))).^2));
188 nonlin020 = 1000*(1-rsq020);
189 alphax020 = 55.3*Ly^2*57/(-c^2*2*T020);
190 alpham020 = 1-sqrt(exp(alphax020));
191 Y020filt = fft(y020filt,Nfft,1);
192 %% 300 filtering 82.1Hz
193 octfilt300 = octaveFilter(80,'1/6 octave');
194 filter300 = octfilt300(dirac);
195 y300filt = flip(octfilt300(flip(y300)));
196 Y300filter = fft(filter300,Nfft);
197 y300filt = y300filt(1:40000);
198 D300 = 10*log10(flip(cumsum(flip(y300filt.^2))));
199 D300 = D300 - max(D300);
200 idD300 = find(D300 <= -5 & D300 >= -25);
201 p300 = polyfit(tvec(idD300),D300(idD300),1);
202 R300 = polyval(p300,tvec);
203 T300 = 60/(R300(1)-R300(end));
204 rsq300 = (sum((R300(idD300)-mean(D300(idD300))).^2)) / ...
        (sum((D300(idD300)-mean(D300(idD300))).^2));
```

```
nonlin300 = 1000*(1-rsq300);
alphax300 = 55.3*Lx^2*82.1/(-c^2*3*T300);
alpham300 = 1-exp(alphax300);
Y300filt = fft(y300filt,Nfft,1);
%% 030 filtering
octfilt030 = octaveFilter(84,'1/6 octave');
filter030 = octfilt030(dirac);
y030filt = flip(octfilt030(flip(y030)));
Y030filter = fft(filter030,Nfft);
y030filt = y030filt(1:24000);
D030 = 10*log10(flip(cumsum(flip(y030filt.^2))));
D030 = D030 - max(D030);
idD030 = find(D030 <= -5 & D030 >= -25);
p030 = polyfit(tvec(idD030),D030(idD030),1);
R030 = polyval(p030,tvec);
T030 = 60/(R030(1)-R030(end));
rsq030 = (sum((R030(idD030)-mean(D030(idD030))).^2)) / ...
    (sum((D030(idD030)-mean(D030(idD030))).^2));
nonlin030 = 1000*(1-rsq030);
alphax030 = 55.3*Ly^2*84.8/(-c^2*3*T030);
alpham030 = 1-sqrt(exp(alphax030));
Y030filt = fft(y030filt,Nfft,1);
%% Plot decay curve lowest 7 modes
figure()
sgtitle('Decay curves and parmeters for room A','FontSize',18)
subplot(3,2,1),plot(D100),title(['100 mode, 28.1 Hz, T_{20} = ...
    ',num2str(T100,3),', \alpha_m = ',num2str(alpham100,3),', and \xi = ...
    ',num2str(nonlin100,3)]),hold on,plot(R100,'--k'),plot([idD100(1) ...
    idD100(end)],D100([idD100(1) idD100(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
subplot(3,2,2),plot(D010),title(['010 mode, 29.6 Hz, T_{20} = ...
    ',num2str(T010,3),', \alpha_m = ',num2str(alpham010,3),', and \xi = ...
    ',num2str(nonlin010,3)]),hold on,plot(R010,'--k'),plot([idD010(1) ...
    idD010(end)],D010([idD010(1) idD010(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
subplot(3,2,3),plot(D200),title(['200 mode, 54.8 Hz, T_{20} = ...
    ',num2str(T200,3),', \alpha_m = ',num2str(alpham200,3),', and \xi = ...
    ',num2str(nonlin200,3)]),hold on,plot(R200,'--k'),plot([idD200(1) ...
    idD200(end)],D200([idD200(1) idD200(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
subplot(3,2,4),plot(D020),title(['020 mode, 57.0 Hz, T_{20} = ...
    ',num2str(T020,3),', \alpha_m = ',num2str(alpham020,3),', and \xi = ...
    ',num2str(nonlin020,3)]),hold on,plot(R020,'--k'),plot([idD020(1) ...
    idD020(end)],D020([idD020(1) idD020(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
subplot(3,2,5),plot(D300),title(['300 mode, 82.1 Hz, T_{20} = ...
    ',num2str(T300,3),', \alpha_m = ',num2str(alpham300,3),', and \xi = ...
    ',num2str(nonlin300,3)]),hold on,plot(R300,'--k'),plot([idD300(1) ...
    idD300(end)],D300([idD300(1) idD300(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
subplot(3,2,6),plot(D030),title(['030 mode, 85.1 Hz, T_{20} = ...
    ',num2str(T030,3),', \alpha_m = ',num2str(alpham030,3),', and \xi = ...
    ',num2str(nonlin030,3)]),hold on,plot(R030,'--k'),plot([idD030(1) ...
    idD030(end)],D030([idD030(1) idD030(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
set(gca, 'fontsize', 18)
%% Plot in frequency
figure()
sgtitle('Frequency responses with source and receiver position cancelling ...
    in room A','FontSize',18)
```

```
239 subplot(3,2,1)
240 plot(fvec(ivf),20*log10(abs(Y100(ivf))),fvec(ivf),20*log10(abs(Y100filter(ivf))))
241 title('100 mode, 28.1 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
242 xline(eigenfreqs(2,1),'--k')
243 xline(AresX(1))
244 ylim([-40 35])
245 legend({'Room''s frequency response','Filter''s frequency response','Rigid ...
        room''s natural frequency'},'Location','best','FontSize',16)
246 xlim([minfreq maxfreq])
247 subplot(3,2,2)
248 plot(fvec(ivf),20*log10(abs(Y010(ivf))),fvec(ivf),20*log10(abs(Y010filter(ivf))))
249 title('010 mode, 29.6 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
250 xline(eigenfreqs(3,1),'--k')
251 xline(AresY(1))
252 ylim([-40 35])
253 xlim([minfreq maxfreq])
254
255 subplot(3,2,3)
256 plot(fvec(ivf),20*log10(abs(Y200(ivf))),fvec(ivf),20*log10(abs(Y200filter(ivf))))
257 title('200 mode, 54.8 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
258 xline(eigenfreqs(6,1),'--k')
259 xline(AresX(2))
260 ylim([-40 35])
261 xlim([minfreq maxfreq])
262 subplot(3,2,4)
263 plot(fvec(ivf),20*log10(abs(Y020(ivf))),fvec(ivf),20*log10(abs(Y020filter(ivf))))
264 title('020 mode, 57.0 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
265 xline(eigenfreqs(7,1),'--k')
266 xline(AresY(2))
267 ylim([-40 35])
268 xlim([minfreq maxfreq])
269
270 subplot(3,2,5)
271 plot(fvec(ivf),20*log10(abs(Y300(ivf))),fvec(ivf),20*log10(abs(Y300filter(ivf))))
272 title('300 mode, 82.1 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
273 xline(eigenfreqs(18,1),'--k')
274 xline(AresX(3))
275 ylim([-40 35])
276 xlim([minfreq maxfreq])
277 subplot(3,2,6)
278 plot(fvec(ivf),20*log10(abs(Y030(ivf))),fvec(ivf),20*log10(abs(Y030filter(ivf))))
279 title('030 mode, 85.1 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
280 xline(eigenfreqs(19,1),'--k')
281 xline(AresY(3))
282 ylim([-40 35])
283 xlim([minfreq maxfreq])
```

Code listing B.14: ModalT_B524.m

```
1 clear
2 close all
3
```

```
Lx = 3.5; % Length
Ly = 2.3; % Width
Lz = 3.4; % Height
T = 0.5; % Reverberation time at low frequencies
c = 340; % speed of air in the room
Nfft = 2^19; % FFT Size
FS = 44100; % Samplerate
maxfreq = 200; % max frequency plotted
minfreq = 20; % min frequency plotted
dims = [Lx Ly Lz];
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft; % frequency vecotr
tvec(:,1) = (1:1*44101)/44100; % time vector for 1 second
ivf = find(fvec < maxfreq); % frequency index vector
eigenfreqs = Shoeboxfreq(dims,fvec(ivf),c);
[~,idf] = sort(eigenfreqs,1);
eigenfreqs = eigenfreqs(idf,:); %Sorted natural frequencies for hard walls
dirac = [1; zeros(136709,1)]; %unit pulse
BresX = [52.8 102 150]; % First axial modal frequencies observed for the X axis
BresY = [82.4 160]; % First axial modal frequencies observed for the Y axis
%% The following section imports the impulse responses
%% LSP
yLSP = audioread('0-180_averageTimeDomain.wav');
YLSP = fft(yLSP(1:7000),Nfft,1);
%% 100
y100 = audioread('B524/ModalT/100.wav');
Y100 = fft(y100(1:T*100000),Nfft,1)./YLSP;
%% 200
y200 = audioread('B524/ModalT/200.wav');
Y200 = fft(y200(1:T*100000),Nfft,1)./YLSP;
%% 300
y300 = audioread('B524/ModalT/300.wav');
Y300 = fft(y300(1:T*100000),Nfft,1)./YLSP;
%% 010
y010 = audioread('B524/ModalT/010.wav');
Y010 = fft(y010(1:T*100000),Nfft,1)./YLSP;
%% The following sections backwards filter the IRs before they are ...
     backwards integrated. Then a least-squares line is calculated and all ...
     the parameters are calculated. Like T_modal, Xi, and alpha.
%% 100 filtering
octfilt100 = octaveFilter(53.1,'2/3 octave');
filter100 = octfilt100(dirac);
y100filt = flip(octfilt100(flip(y100)));
Y100filter = fft(filter100,Nfft);
y100filt = y100filt(1:16500);
D100 = 10*log10(flip(cumsum(flip(y100filt.^2))));
D100 = D100-max(D100);
idD100 = find(D100 <= -5 & D100 >= -25);
p100 = polyfit(tvec(idD100),D100(idD100),1);
R100 = polyval(p100,tvec);
T100 = 60/(R100(1)-R100(end));
rsq100 = (sum((R100(idD100)-mean(D100(idD100))).^2)) / ...
     (sum((D100(idD100)-mean(D100(idD100))).^2));
nonlin100 = 1000*(1-rsq100);
alphax100 = 55.3*Lx^2*52.8/(-c^2*1*T100);
alpham100 = 1-sqrt(exp(alphax100));
Y100filt = fft(y100filt,Nfft,1);
%% 200 filtering
octfilt200 = octaveFilter(100,'1/3 octave');
filter200 = octfilt200(dirac);
y200filt = flip(octfilt200(flip(y200)));
```

```
Y200filter = fft(filter200,Nfft);
y200filt = y200filt(1:7000);
D200 = 10*log10(flip(cumsum(flip(y200filt.^2))));
D200 = D200-max(D200);
idD200 = find(D200 <= -5 & D200 >= -25);
p200 = polyfit(tvec(idD200),D200(idD200),1);
R200 = polyval(p200,tvec);
T200 = 60/(R200(1)-R200(end));
rsq200 = (sum((R200(idD200)-mean(D200(idD200))).^2)) / ...
    (sum((D200(idD200)-mean(D200(idD200))).^2));
nonlin200 = 1000*(1-rsq200);
alphax200 = 55.3*Lx^2*102/(-c^2*2*T200);
alpham200 = 1-sqrt(exp(alphax200));
Y200filt = fft(y200filt,Nfft,1);
%% 300 filtering 154 Hz
octfilt300 = octaveFilter(148,'1/3 octave');
filter300 = octfilt300(dirac);
y300filt = flip(octfilt300(flip(y300)));
Y300filter = fft(filter300,Nfft);
y300filt = y300filt(1:19000);
D300 = 10*log10(flip(cumsum(flip(y300filt.^2))));
D300 = D300 - max(D300);
idD300 = find(D300 <= -5 & D300 >= -25);
p300 = polyfit(tvec(idD300),D300(idD300),1);
R300 = polyval(p300,tvec);
T300 = 60/(R300(1)-R300(end));
rsq300 = (sum((R300(idD300)-mean(D300(idD300))).^2)) / ...
    (sum((D300(idD300)-mean(D300(idD300))).^2));
nonlin300 = 1000*(1-rsq300);
alphax300 = 55.3*Lx^2*150/(-c^2*3*T300);
alpham300 = 1-sqrt(exp(alphax300));
Y300filt = fft(y300filt,Nfft,1);
%% 010 filtering
octfilt010 = octaveFilter(82.6,'1/3 octave');
filter010 = octfilt010(dirac);
y010filt = flip(octfilt010(flip(y010)));
Y010filter = fft(filter010,Nfft);
y010filt = y010filt(1:16000);
D010 = 10*log10(flip(cumsum(flip(y010filt.^2))));
D010 = D010-max(D010);
idD010 = find(D010 <= -5 & D010 >= -25);
p010 = polyfit(tvec(idD010),D010(idD010),1);
R010 = polyval(p010,tvec);
T010 = 60/(R010(1)-R010(end));
rsq010 = (sum((R010(idD010)-mean(D010(idD010))).^2)) / ...
    (sum((D010(idD010)-mean(D010(idD010))).^2));
nonlin010 = 1000*(1-rsq010);
alpham010 = 1-exp(13.8*Ly/(-c*T010));
Y010filt = fft(y010filt,Nfft,1);
%% Plot decay curve lowest 7 modes
figure()
sgtitle('Decay curves and parmeters for room B','FontSize',18)
subplot(3,2,1),plot(D100),title(['100 mode, 52.8 Hz, T_{20} = ...
    ',num2str(T100,3),', \alpha_m = ',num2str(alpham100,3),', and \xi = ...
    ',num2str(nonlin100,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
    on,plot(R100,'--k'),plot([idD100(1) idD100(end)],D100([idD100(1) ...
    idD100(end)]),'or'),hold off,ylim([-60 ...
    0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
subplot(3,2,2),plot(D010),title(['010 mode, 82.4 Hz, T_{20} = ...
    ',num2str(T010,3),', \alpha_m = ',num2str(alpham010,3),', and \xi = ...
```

```
        ',num2str(nonlin010,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
        on,plot(R010,'--k'),plot([idD010(1) idD010(end)],D010([idD010(1) ...
        idD010(end)]),'or'),hold off,ylim([-60 ...
        0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
112 subplot(3,2,3),plot(D200),title(['200 mode, 102.0 Hz, T_{20} = ...
        ',num2str(T200,3),', \alpha_m = ',num2str(alpham200,3),', and \xi = ...
        ',num2str(nonlin200,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
        on,plot(R200,'--k'),plot([idD200(1) idD200(end)],D200([idD200(1) ...
        idD200(end)]),'or'),hold off,ylim([-60 ...
        0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
113 subplot(3,2,5),plot(D300),title(['300 mode, 150.0 Hz, T_{20} = ...
        ',num2str(T300,3),', \alpha_m = ',num2str(alpham300,3),', and \xi = ...
        ',num2str(nonlin300,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
        on,plot(R300,'--k'),plot([idD300(1) idD300(end)],D300([idD300(1) ...
        idD300(end)]),'or'),hold off,ylim([-60 ...
        0]),xlabel('Samples'),ylabel('[dB]'),set(gca, 'fontsize', 15)
114 set(gca, 'fontsize', 18)
115 %% Plot in frequency
116 figure()
117 sgtitle('Frequency responses with source and receiver position cancelling ...
        in room B','FontSize',18)
118 subplot(3,2,1)
119 plot(fvec(ivf),20*log10(abs(Y100(ivf))),fvec(ivf),20*log10(abs(Y100filter(ivf))))
120 title('100 mode, 52.8 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
121 xline(eigenfreqs(2,1),'--k')
122 xline(BresX(1))
123 %legend({'Room''s frequency response','Filter''s frequency ...
        response','Rigid room''s natural frequency'},'Location','best')
124 ylim([-40 30])
125 xlim([minfreq maxfreq])
126 subplot(3,2,2)
127 plot(fvec(ivf),20*log10(abs(Y010(ivf))),fvec(ivf),20*log10(abs(Y010filter(ivf))))
128 title('010 mode, 82.4 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
129 xline(eigenfreqs(5,1),'--k')
130 xline(BresY(1))
131 ylim([-40 30])
132 xlim([minfreq maxfreq])
133 subplot(3,2,3)
134 plot(fvec(ivf),20*log10(abs(Y200(ivf))),fvec(ivf),20*log10(abs(Y200filter(ivf))))
135 title('200 mode, 102.0 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
136 xline(eigenfreqs(8,1),'--k')
137 xline(BresX(2))
138 ylim([-40 30])
139 xlim([minfreq maxfreq])
140 subplot(3,2,5)
141 plot(fvec(ivf),20*log10(abs(Y300(ivf))),fvec(ivf),20*log10(abs(Y300filter(ivf))))
142 title('300 mode, 150.0 Hz'),xlabel('Frequency ...
        [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 15)
143 xline(eigenfreqs(18,1),'--k')
144 xline(BresX(3))
145 ylim([-40 30])
146 xlim([minfreq maxfreq])
```

Code listing B.15: ModalT_B553.m

```
clear
close all

Lx = 3.05; %Length
Ly = 2.3; %Width
Lz = 3.4; %Height
T = 0.5; % Reverberation time at low frequencies
c = 340; % speed of air in the room
Nfft = 2^19; % FFT Size
FS = 44100; % Samplerate
maxfreq = 200; % max frequency plotted
minfreq = 20; % min frequency plotted
dims = [Lx Ly Lz];
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft; % frequency vecotr
tvec(:,1) = (1:1*44101)/44100; % time vector for 1 second
ivf = find(fvec < maxfreq); % frequency index vectoreigenfreqs = ...
    Shoeboxfreq(dims,fvec(ivf),c);
eigenfreqs = Shoeboxfreq(dims,fvec(ivf),c);
[~,idf] = sort(eigenfreqs,1);
eigenfreqs = eigenfreqs(idf,:);
dirac = [1; zeros(136709,1)]; %unit pulse
CresX = [61.7 118 174.5];% First axial modal frequencies observed for the ...
    X axis
CresY = [78.7 154];% First axial modal frequencies observed for the Y axis
%% The following section imports the impulse responses
%% LSP
yLSP = audioread('0-180_averageTimeDomain.wav');
YLSP = fft(yLSP(1:7000),Nfft,1);
%% 100
y100 = audioread('B553/ModalT/100.wav');
Y100 = fft(y100(1:T*100000),Nfft,1)./YLSP;
%% 200
y200 = audioread('B553/ModalT/200.wav');
Y200 = fft(y200(1:T*100000),Nfft,1)./YLSP;
%% 300
y300 = audioread('B553/ModalT/300.wav');
Y300 = fft(y300(1:T*100000),Nfft,1)./YLSP;
%% 010
y010 = audioread('B553/ModalT/010.wav');
Y010 = fft(y010(1:T*100000),Nfft,1)./YLSP;
%% 020
y020 = audioread('B553/ModalT/020.wav');
Y020 = fft(y020(1:T*100000),Nfft,1)./YLSP;
%% The following sections backwards filter the IRs before they are ...
    backwards integrated. Then a least-squares line is calculated and all ...
    the parameters are calculated. Like T_modal, Xi, and alpha.
%% 100 filtering
octfilt100 = octaveFilter(61.15,'2/3 octave');
filter100 = octfilt100(dirac);
y100filt = octfilt100(y100);
Y100filter = fft(filter100,Nfft);
y100filt = y100filt(1:16000);
D100 = 10*log10(flip(cumsum(flip(y100filt.^2))));
D100 = D100-max(D100);
idD100 = find(D100 <= -5 & D100 >= -25);
p100 = polyfit(tvec(idD100),D100(idD100),1);
R100 = polyval(p100,tvec);
T100 = 60/(R100(1)-R100(end));
rsq100 = (sum((R100(idD100)-mean(D100(idD100))).^2)) / ...
    (sum((D100(idD100)-mean(D100(idD100))).^2));
```

```
nonlin100 = 1000*(1-rsq100);
alphax100 = 55.3*Lx^2*61.7/(-c^2*1*T100);
alpham100 = 1-sqrt(exp(alphax100));
Y100filt = fft(y100filt,Nfft,1);
%% 200 filtering
octfilt200 = octaveFilter(117.9,'1/6 octave');
filter200 = octfilt200(dirac);
y200filt = flip(octfilt200(flip(y200)));
Y200filter = fft(filter200,Nfft);
y200filt = y200filt(1:24000);
D200 = 10*log10(flip(cumsum(flip(y200filt.^2))));
D200 = D200-max(D200);
idD200 = find(D200 <= -5 & D200 >= -25);
p200 = polyfit(tvec(idD200),D200(idD200),1);
R200 = polyval(p200,tvec);
T200 = 60/(R200(1)-R200(end));
rsq200 = (sum((R200(idD200)-mean(D200(idD200))).^2)) / ...
    (sum((D200(idD200)-mean(D200(idD200))).^2));
nonlin200 = 1000*(1-rsq200);
alphax200 = 55.3*Lx^2*118/(-c^2*2*T200);
alpham200 = 1-sqrt(exp(alphax200));
Y200filt = fft(y200filt,Nfft,1);
%% 300 filtering
octfilt300 = octaveFilter(174.4,'1/12 octave');
filter300 = octfilt300(dirac);
y300filt = flip(octfilt300(flip(y300)));
Y300filter = fft(filter300,Nfft);
y300filt = y300filt(1:19000);
D300 = 10*log10(flip(cumsum(flip(y300filt.^2))));
D300filter = 10*log10(flip(cumsum(flip(filter300.^2))));
D300 = D300 - max(D300);
idD300 = find(D300 <= -5 & D300 >= -25);
p300 = polyfit(tvec(idD300),D300(idD300),1);
R300 = polyval(p300,tvec);
T300 = 60/(R300(1)-R300(end));
rsq300 = (sum((R300(idD300)-mean(D300(idD300))).^2)) / ...
    (sum((D300(idD300)-mean(D300(idD300))).^2));
nonlin300 = 1000*(1-rsq300);
alphax300 = 55.3*Lx^2*174.5/(-c^2*3*T300);
alpham300 = 1-sqrt(exp(alphax300));
Y300filt = fft(y300filt,Nfft,1);
%% 010 filtering
octfilt010 = octaveFilter(78.4,'1/3 octave');
filter010 = octfilt010(dirac);
y010filt = flip(octfilt010(flip(y010)));
Y010filter = fft(filter010,Nfft);
y010filt = y010filt(1:14000);
D010 = 10*log10(flip(cumsum(flip(y010filt.^2))));
D010 = D010-max(D010);
idD010 = find(D010 <= -5 & D010 >= -25);
p010 = polyfit(tvec(idD010),D010(idD010),1);
R010 = polyval(p010,tvec);
T010 = 60/(R010(1)-R010(end));
rsq010 = (sum((R010(idD010)-mean(D010(idD010))).^2)) / ...
    (sum((D010(idD010)-mean(D010(idD010))).^2));
nonlin010 = 1000*(1-rsq010);
alphax010 = 55.3*Ly^2*78.7/(-c^2*1*T010);
alpham010 = 1-sqrt(exp(alphax010));
Y010filt = fft(y010filt,Nfft,1);
%% 020 filtering
```

```
octfilt020 = octaveFilter(154.2,'1/6 octave');
filter020 = octfilt020(dirac);
y020filt = octfilt020(y020);
Y020filter = fft(filter020,Nfft);
y020filt = y020filt(1:20000);
D020 = 10*log10(flip(cumsum(flip(y020filt.^2))));
D020 = D020-max(D020);
idD020 = find(D020 <= -5 & D020 >= -25);
p020 = polyfit(tvec(idD020),D020(idD020),1);
R020 = polyval(p020,tvec);
T020 = 60/(R020(1)-R020(end));
rsq020 = (sum((R020(idD020)-mean(D020(idD020))).^2)) / ...
    (sum((D020(idD020)-mean(D020(idD020))).^2));
nonlin020 = 1000*(1-rsq020);
alphax020 = 55.3*Ly^2*154/(-c^2*2*T020);
alpham020 = 1-sqrt(exp(alphax020));
Y020filt = fft(y020filt,Nfft,1);
%% Plot decay curve lowest 7 modes
figure()
sgtitle('Decay curves and parmeters for room C','FontSize',18)
subplot(3,2,1),plot(D100),title(['100 mode, 61.7 Hz, T_{20} = ...
    ',num2str(T100,3),', \alpha_m = ',num2str(alpham100,3),', and \xi = ...
    ',num2str(nonlin100,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
    on,plot(R100,'--k'),plot([idD100(1) idD100(end)],D100([idD100(1) ...
    idD100(end)]),'or'),hold off,ylim([-60 0]),set(gca, 'fontsize', 17)
subplot(3,2,2),plot(D010),title(['010 mode, 78.7 Hz, T_{20} = ...
    ',num2str(T010,3),', \alpha_m = ',num2str(alpham010,3),', and \xi = ...
    ',num2str(nonlin010,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
    on,plot(R010,'--k'),plot([idD010(1) idD010(end)],D010([idD010(1) ...
    idD010(end)]),'or'),hold off,ylim([-60 0]),set(gca, 'fontsize', 17)
subplot(3,2,3),plot(D200),title(['200 mode, 118.0 Hz, T_{20} = ...
    ',num2str(T200,3),', \alpha_m = ',num2str(alpham200,3),', and \xi = ...
    ',num2str(nonlin200,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
    on,plot(R200,'--k'),plot([idD200(1) idD200(end)],D200([idD200(1) ...
    idD200(end)]),'or'),hold off,ylim([-60 0]),set(gca, 'fontsize', 17)
subplot(3,2,4),plot(D020),title(['020 mode, 154.0 Hz, T_{20} = ...
    ',num2str(T020,3),', \alpha_m = ',num2str(alpham020,3),', and \xi = ...
    ',num2str(nonlin020,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
    on,plot(R020,'--k'),plot([idD020(1) idD020(end)],D020([idD020(1) ...
    idD020(end)]),'or'),hold off,ylim([-60 0]),set(gca, 'fontsize', 17)
subplot(3,2,5),plot(D300),title(['300 mode, 174.5 Hz, T_{20} = ...
    ',num2str(T300,3),', \alpha_m = ',num2str(alpham300,3),', and \xi = ...
    ',num2str(nonlin300,3)]),xlabel('Samples'),ylabel('[dB]'),hold ...
    on,plot(R300,'--k'),plot([idD300(1) idD300(end)],D300([idD300(1) ...
    idD300(end)]),'or'),hold off,ylim([-60 0]),set(gca, 'fontsize', 17)
%% Plot in frequency
figure()
sgtitle('Frequency responses with source and receiver position cancelling ...
    in room C','FontSize',18)
subplot(3,2,1)
plot(fvec(ivf),20*log10(abs(Y100(ivf))),fvec(ivf),20*log10(abs(Y100filter(ivf))))
title('100 mode, 61.7 Hz'),xlabel('Frequency ...
    [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 18)
xline(eigenfreqs(3,1),'--k')
xline(CresX(1))
%legend({'Room''s frequency response','Filter''s frequency ...
    response','Rigid room''s natural frequency'},'Location','best')
ylim([-40 30])
xlim([minfreq maxfreq])
subplot(3,2,2)
```

```
plot(fvec(ivf),20*log10(abs(Y010(ivf))),fvec(ivf),20*log10(abs(Y010filter(ivf))))
title('010 mode, 78.7 Hz'),xlabel('Frequency ...
    [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 18)
xline(eigenfreqs(4,1),'--k')
xline(CresY(1))
ylim([-40 30])
xlim([minfreq maxfreq])
subplot(3,2,3)
plot(fvec(ivf),20*log10(abs(Y200(ivf))),fvec(ivf),20*log10(abs(Y200filter(ivf))))
title('200 mode, 118.0 Hz'),xlabel('Frequency ...
    [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 18)
xline(eigenfreqs(10,1),'--k')
xline(CresX(2))
ylim([-40 30])
xlim([minfreq maxfreq])
subplot(3,2,4)
plot(fvec(ivf),20*log10(abs(Y020(ivf))),fvec(ivf),20*log10(abs(Y020filter(ivf))))
title('020 mode, 154.0 Hz'),xlabel('Frequency ...
    [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 18)
xline(eigenfreqs(17,1),'--k')
xline(CresY(2))
ylim([-40 30])
xlim([minfreq maxfreq])
subplot(3,2,5)
plot(fvec(ivf),20*log10(abs(Y300(ivf))),fvec(ivf),20*log10(abs(Y300filter(ivf))))
title('300 mode, 174.5 Hz'),xlabel('Frequency ...
    [Hz]'),ylabel('[dB]'),set(gca, 'fontsize', 18)
xline(eigenfreqs(25,1),'--k')
xline(CresX(3))
ylim([-40 30])
xlim([minfreq maxfreq])
```

Code listing B.16: absorptions.m

```
%% init
clear
close all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script plots all the estimated absorption coefficients.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% vars
Nfft = 2^18;
FS = 44100;
fvec(:,1) = FS*(0:(Nfft/2-1))/Nfft;
omega = 2*pi*fvec;
c0 = 339;

% Absorption coefficient of walls mostly constructed in glass
glass.RoomAalpha = [0.366 0.204 0.557];
glass.RoomAalphaM = [0.249 0.244 0.549];
glass.RoomAalphaTmBad = [0.421, -1, -1];
glass.RoomAalphaTm = [-1 0.152 0.247];
glass.RoomAivf = [177 340 505];
glass.RoomBalpha = [0.322 0.307];
glass.RoomBalphaM = [0.014 0.565];
glass.RoomBivf = [491 952];
glass.RoomCalpha = [0.359 0.190 -1];
glass.RoomCalphaM = [0.212 0.014 -1];
```

```
glass.RoomCalphaTm = [0.315 0.223 0.26];
glass.RoomCivf = [368 703 2076];

% Absorption coefficient of gypsum walls.
gypsium.RoomAalpha = [0.226 0.242 -1];
gypsium.RoomAalphaBad = [-1 -1 0.616];
gypsium.RoomAalphaM = [0.01 0.347 0.728];
gypsium.RoomAalphaTm = [0.337 0.349 0.441];
gypsium.RoomAivf = [168 327 483];
gypsium.RoomBalpha = [0.424 0.312 0.312];
gypsium.RoomBalphaM = [0.013 0.018 0.457];
gypsium.RoomBalphaTmBad = [-1 0.797 0.253];
gypsium.RoomBalphaTm = [0.359 -1 -1];
gypsium.RoomBivf = [315 607 893];
gypsium.RoomCalpha = [0.325 -1];
gypsium.RoomCalphaBad = [-1 0.693];
gypsium.RoomCalphaM = [0.025 0.789];
gypsium.RoomCalphaTmBad = [-1 0.225];
gypsium.RoomCalphaTm = [0.294 -1];
gypsium.RoomCivf = [469 917];

figure()
sgtitle('Absorption Coefficient')
subplot(1,2,2)
hold on
plot(fvec(glass.RoomAivf),glass.RoomAalpha,'ob','DisplayName','Room A, ...
    standing wave method','MarkerSize',15)
plot(fvec(glass.RoomAivf),glass.RoomAalphaM,'+b','DisplayName','Room A, ...
    standing wave model','MarkerSize',5)
plot(fvec(glass.RoomAivf),glass.RoomAalphaTm,'*b','DisplayName','Room A, ...
    modal reverberation time method','MarkerSize',15)
plot(fvec(glass.RoomAivf),glass.RoomAalphaTmBad,'*b','HandleVisibility','off','MarkerSize',5)
plot(fvec(glass.RoomBivf),glass.RoomBalpha,'om','DisplayName','Room B, ...
    standing wave method','MarkerSize',15)
plot(fvec(glass.RoomBivf),glass.RoomBalphaM,'+m','DisplayName','Room B, ...
    standing wave model','MarkerSize',5)
plot(fvec(glass.RoomCivf),glass.RoomCalpha,'or','DisplayName','Room C, ...
    standing wave method','MarkerSize',15)
plot(fvec(glass.RoomCivf),glass.RoomCalphaM,'+r','DisplayName','Room C, ...
    standing wave model','MarkerSize',5)
plot(fvec(glass.RoomCivf),glass.RoomCalphaTm,'*r','DisplayName','Room C, ...
    modal reverberation time method','MarkerSize',15)
hold off
title('Glass wall'),xlabel('Frequency [Hz]')
xlim([20 200])
ylim([0 1])
set(gca, 'fontsize', 18)
grid minor

subplot(1,2,1)
hold on
plot(fvec(gypsium.RoomAivf),gypsium.RoomAalpha,'ob','DisplayName','Room A, ...
    standing wave method','MarkerSize',15)
plot(fvec(gypsium.RoomAivf),gypsium.RoomAalphaBad,'ob','HandleVisibility','off','MarkerSize',5)
plot(fvec(gypsium.RoomAivf),gypsium.RoomAalphaM,'+b','DisplayName','Room ...
    A, standing wave model','MarkerSize',5)
plot(fvec(gypsium.RoomAivf),gypsium.RoomAalphaTm,'*b','DisplayName','Room ...
    A, modal reverberation time method','MarkerSize',15)
plot(fvec(gypsium.RoomBivf),gypsium.RoomBalpha,'om','DisplayName','Room B, ...
    standing wave method','MarkerSize',15)
```

```
plot(fvec(gypsium.RoomBivf),gypsium.RoomBalphaM,'+m','DisplayName','Room ...
    B, standing wave model','MarkerSize',5)
plot(fvec(gypsium.RoomBivf),gypsium.RoomBalphaTm,'*m','DisplayName','Room ...
    B, modal reverberation time method','MarkerSize',15)
plot(fvec(gypsium.RoomBivf),gypsium.RoomBalphaTmBad,'*m','HandleVisibility','off','MarkerSize',5)
plot(fvec(gypsium.RoomCivf),gypsium.RoomCalpha,'or','DisplayName','Room C, ...
    standing wave method','MarkerSize',15)
plot(fvec(gypsium.RoomCivf),gypsium.RoomCalphaBad,'or','HandleVisibility','off','MarkerSize',5)
plot(fvec(gypsium.RoomCivf),gypsium.RoomCalphaM,'+r','DisplayName','Room ...
    C, standing wave model','MarkerSize',5)
plot(fvec(gypsium.RoomCivf),gypsium.RoomCalphaTm,'*r','DisplayName','Room ...
    C, modal reverberation time method','MarkerSize',15)
plot(fvec(gypsium.RoomCivf),gypsium.RoomCalphaTmBad,'*r','HandleVisibility','off','MarkerSize',5)
hold off
title('Gypsum wall'),xlabel('Frequency [Hz]')
legend('Location','best')
xlim([20 200])
ylim([0 1])
set(gca, 'fontsize', 18)
grid minor
```

Code listing B.17: wall.m

```
classdef wall < handle
  % This script calculates the rough estimate for the wall impedance and
  % absorption coefficient at low frequencies.
  % FUNCTIONS:
  %   wall: Calculates the plate's parameters based off the material ...
      type, its
  %         thickness and the dimensions of the plate.
  %
  % Author : Henrik K. R. Berg
  % E-mail : hkrb94@gmail.com

  properties
      type % material type
      rho % density
      lx % width of plate
      ly % height of plate
      h % thickness of plate
      m % mass of plate
      poirat % Poisson's ratio
      loss % Total lossfactor
      intloss % Initial loss factor
      E % Young's modulus
      B % Bending Stiffness
      fc % Critical frequency
      c0 = 340; % Speed of air in the room
      f11 % Fundmental structural mode of the room
      alpha % Absorption coefficient of the wall
      salpha = 0.1 % Assumed structural absorption coefficient of edge ...
          conncetions
      Zm % Wall impedance
      Rp % Reflection factor
      fvec = 44100*(0:((2^18)/2-1))/(2^18); % Frequency vector with Nfft ...
          = 2^18
  end
  methods
```

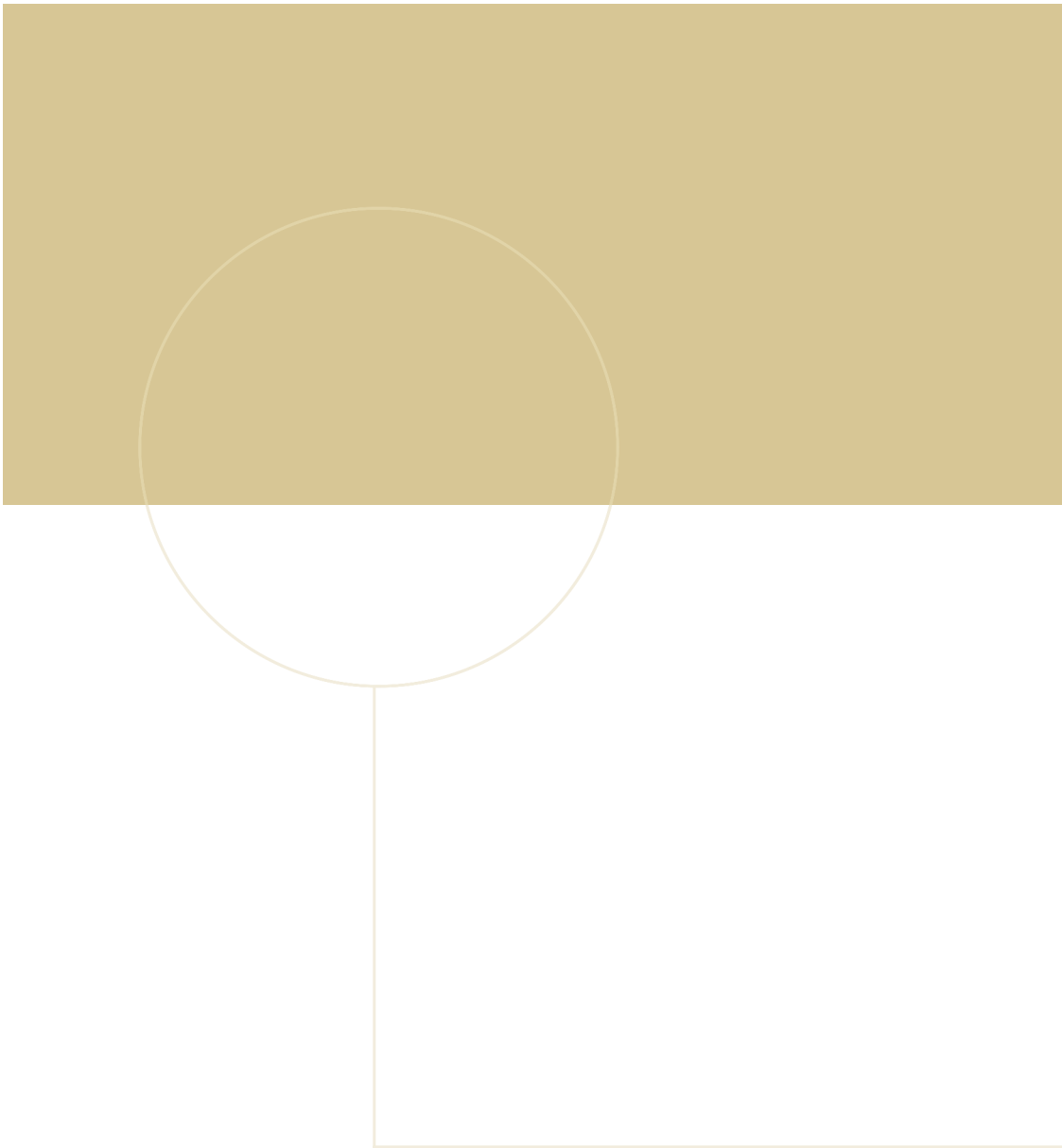
```
33          function obj = wall(material,h,lx,ly) %constructor
34              if isa(material,'char')
35                  material = convertCharsToStrings(material);
36              elseif isnumeric(material)
37                  disp('**********************************************************************************')
38                  disp('Please enter type of material in lower case letters: ...
                        valid inputs are:')
39                  disp('''glass''')
40                  disp('''gypsum''')
41                  disp('**********************************************************************************')
42                  return
43              elseif isa(material,'string')
44              else
45                  disp('**********************************************************************************')
46                  disp('Please enter type of material in lower case letters: ...
                        valid inputs are')
47                  disp('''glass''')
48                  disp('''gypsum''')
49                  disp('**********************************************************************************')
50                  return
51              end
52              if material == "glass"
53                      obj.type = "glass";
54                      obj.rho = 2500;
55                      obj.E = 60*10^9;
56                      obj.poirat = 0.2;
57                      obj.intloss = 0.002;
58                  elseif material == "gypsum"
59                      obj.type = "gypsum";
60                      obj.rho = 900;
61                      obj.E = 4.1*10^9;
62                      obj.poirat = 0.3;
63                      obj.intloss = 0.015;
64              end
65              obj.lx = lx;
66              obj.ly = ly;
67              obj.h = h;
68              obj.m = obj.rho*obj.h;
69              if material == "gypsum"
70                  obj.B = obj.E*0.0125^3/(12*(1-obj.poirat^2));
71                  obj.fc = obj.c0^2/(2*pi)*sqrt((obj.rho*0.0125)/obj.B);
72              else
73                  obj.B = obj.E*obj.h^3/(12*(1-obj.poirat^2));
74                  obj.fc = obj.c0^2/(2*pi)*sqrt(obj.m/obj.B);
75              end
76              omega = 2*pi*obj.fvec;
77              obj.loss = obj.intloss + ...
                     ((2*obj.lx+2*obj.ly)*obj.c0*obj.salpha)/(pi^2*obj.lx*obj.ly) ...
                     * (obj.fvec*obj.fc).^(-1/2)+(2*1.19*obj.c0)./(omega*obj.m);
78              obj.f11 = obj.c0^2/(4*obj.fc)*((1/obj.lx)^2+(1/obj.ly)^2);
79              obj.Zm = ...
                     1i*omega*obj.m.*(1-(obj.f11./obj.fvec).^2)+obj.loss.*omega*obj.m;
80              obj.Rp = ((obj.Zm/408)-1)./((obj.Zm/408)+1);
81              obj.alpha = 1-abs(obj.Rp).^2;
82          end
83
84          function vals = fn(obj,nx,ny)
85              vals = obj.c0^2/(4*obj.fc)*((nx/obj.lx)^2+(ny/obj.ly)^2);
86              disp(['f(nx,ny) = ',num2str(vals),' Hz'])
87          end
```

```
    end
end
```

Code listing B.18: WallPlots.m

```
clear
close all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script plots the rough estimates of the different plate's absorption
% coefficients. It calls the class "wall" to return objects containing the
% parameters.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Gypsum60 = wall("gypsum",0.025,2.7,0.6);
Gypsum90 = wall("gypsum",0.025,2.7,0.9);
Glass1_48 = wall("glass",0.00636,1.97,1.48);
Glass0_91 = wall("glass",0.00636,1.97,0.915);
thickGlass1_48 = wall("glass",0.00836,1.97,1.48);
thicksmallglass1_48 =  wall("glass",0.00636,0.603,1.48);
smallglass1_48 =  wall("glass",0.00636,0.603,1.48);
smallglass0_91 = wall("glass",0.00636,0.603,0.915);
overdoorglass = wall("glass",0.00636,0.83,0.58);
thickoverdoorglass = wall("glass",0.00636,0.83,0.58);

fvec = Gypsum60.fvec;
ivf = fvec<200 & fvec>20;

AresX = [28.1 54.8 82.1];
AresY = [29.6 57 84.8];
BresX = [52.8 102 150];
BresY = [82.4 160];
CresX = [61.7 118];
CresY = [78.7 154];

%%
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('\alpha from 25mm thick gypsum wall')
plot(fvec(ivf),Gypsum60.alpha(ivf),'DisplayName','Room A, 60 cm stud distance')
hold on
xline(28.1,'-k','DisplayName','Axial modes location')
xline(54.8,'HandleVisibility','off')
xline(82.1,'HandleVisibility','off')
hold off
legend
ylim([0 1])
xlabel('Frequency [Hz]')
set(gca, 'fontsize', 14)
grid minor
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('\alpha from 25mm thick gypsum wall')
hold on
plot(fvec(ivf),Gypsum90.alpha(ivf),'DisplayName','Room B & C, 90 cm stud ...
    distance')
xline(52.8,'-k','DisplayName','Axial modes location, Room B')
xline(102,'HandleVisibility','off')
xline(150,'HandleVisibility','off')
```

```
xline(78.7,'-b','DisplayName','Axial modes location, Room C')
xline(154,'-b','HandleVisibility','off')
hold off
legend
ylim([0 1])
xlabel('Frequency [Hz]')
set(gca, 'fontsize', 14)
grid minor
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('\alpha from 6.36mm thick glass panes, room B')
plot(fvec(ivf),Glass1_48.alpha(ivf),'DisplayName','1.97m x 1.48m glass')
hold on
plot(fvec(ivf),smallglass1_48.alpha(ivf),'DisplayName','0.60m x 1.48m glass')
plot(fvec(ivf),overdoorglass.alpha(ivf),'DisplayName','0.83m x 0.58m glass')
xline(82.4,'-k','DisplayName','Axial modes location')
xline(160,'HandleVisibility','off')
hold off
legend
ylim([0 1])
xlabel('Frequency [Hz]')
set(gca, 'fontsize', 14)
grid minor
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('\alpha from 8.36mm thick glass panes, room A')
plot(fvec(ivf),thickGlass1_48.alpha(ivf),'DisplayName','1.97m x 1.48m glass')
hold on
plot(fvec(ivf),thicksmallglass1_48.alpha(ivf),'DisplayName','0.60m x 1.48m ...
    glass')
plot(fvec(ivf),thickoverdoorglass.alpha(ivf),'DisplayName','0.83m x 0.58m ...
    glass')
xline(29.6,'-k','DisplayName','Axial modes location')
xline(57,'HandleVisibility','off')
xline(85.1,'HandleVisibility','off')
hold off
legend
ylim([0 1])
xlabel('Frequency [Hz]')
set(gca, 'fontsize', 14)
grid minor
figure()
set(gcf,'position',[0,0,1800,600])
sgtitle('\alpha from 6.36mm thick glass panes, room C')
plot(fvec(ivf),Glass0_91.alpha(ivf),'DisplayName','1.97m x 0.91m glass')
hold on
plot(fvec(ivf),smallglass0_91.alpha(ivf),'DisplayName','0.60m x 0.91m glass')
plot(fvec(ivf),overdoorglass.alpha(ivf),'DisplayName','0.83m x 0.58m glass')
xline(61.7,'-k','DisplayName','Axial modes location')
xline(118,'HandleVisibility','off')
xline(174.5,'HandleVisibility','off')
hold off
legend
ylim([0 1])
xlabel('Frequency [Hz]')
set(gca, 'fontsize', 14)
grid minor
% figure()
% figure()
% figure()
```



NTNU

Norwegian University of
Science and Technology