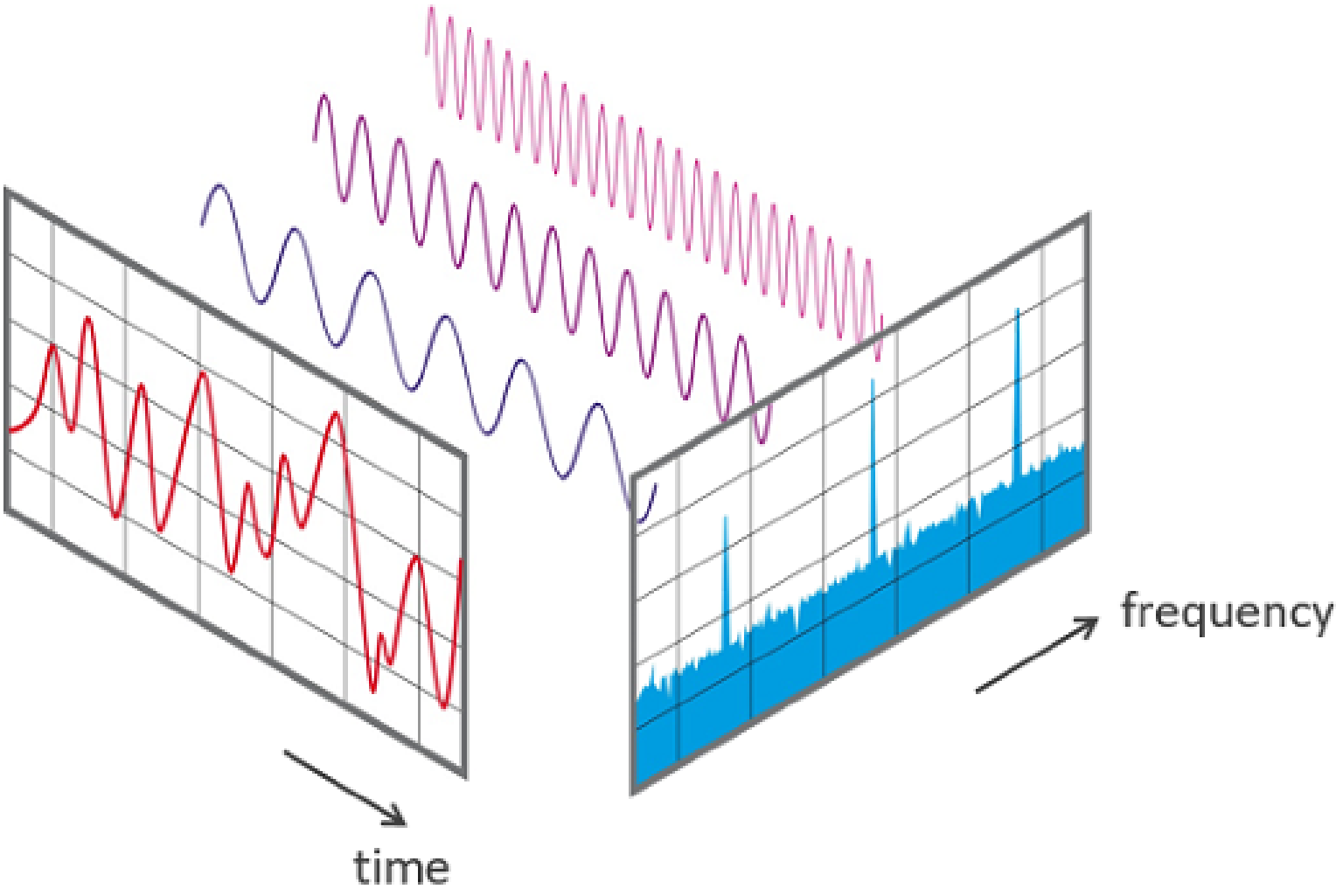
Developing a Multi-Microphone Impedancetube Focused on Determining Room Acoustic Material Properties at Low-Frequencies

Bachelor Project ESD6
Christian Lykke Jørgensen



Department of Electronic Systems
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Impedancetube

Project:
6th Semester Project

Project Period:
Spring 2025

Project Group:
12

Participants:
Christian Lykke Jørgensen

Supervisor:
Flemming Christensen

Page Numbers: 232

Date of Completion:
May 21, 2025

Abstract:

Lorem Ipsum

Contents

Preface

This report searches the requirements to develop an impedance tube based upon DS/EN ISO 10534-2:2023. In appendix B on page 112 all formulas and variables will be noted, in order of appearance.

Aalborg University, May 21, 2025

Christian Lykke Jørgensen
cljo22@student.aau.dk

1 Introduction

Acoustics are present in every aspect of life, no matter how small or large, resulting in acoustics significantly impacting the very perception of life. Scientific acoustical studies began emerging as far back as the 6th century BC with Greek philosophers and later Roman architects/engineers embracing acoustic properties in construction, [1]. However, the contemporary understanding of acoustics has only existed for approximately 200 years, [1]. Within those 200 years, acoustics has changed from a phenomenon only understood by scholars to an aspect of life, that most can relate to or know of to some degree.

In modern construction, acoustical properties have traversed from being reserved for performance centers, to being integrated into almost anything. It has become crucial for the average homeowner to achieve "good acoustics", along with acoustics occupying an increasingly large facet of large-scale construction. To achieve "good acoustics" a multitude of methods are available, ranging from adding a thick carpet or heavy drapes to bass-traps, vibration minimizing, and similar sound-deadening techniques.

Unfortunately, creating an acoustically optimal room is expensive, and in many cases wouldn't resemble a habitable room in common sense. Therefore the next-best option is adding unnoticeable elements such as padded furniture, rugs, pillows, drapes, and acoustic art/images or incorporating constructional options such as floor dampening, carpets, acoustic ceilings, and other, more intrusive options, [2, 3].

To bridge the gap between the historical and practical perspectives on acoustics, it is essential to consider the challenges and trade-offs involved in achieving optimal acoustical environments. While the importance of acoustics in everyday life and construction is evident, the practical implementation of "good acoustics" is often constrained by cost, aesthetics, and feasibility. This raises a fundamental question that initializes this project:

"How can "good acoustics" be defined and quantified in a way that balances theoretical ideals with real-world applicability?"

2 Problem Analysis

2.1 Room Acoustics

Determining good acoustics is as subjective as determining which music is pleasant. While subjective qualities are present, objective aspects should be considered as well, as the very shape of a person (ears, head, shoulders, etc.) influence the perception of sound as well, [4, 5, 6, 7, 8]. Furthermore, good acoustical properties are not as "simple" as attempting to achieve anechoic conditions, as the inherent psychoacoustical properties of an anechoic room are not considered pleasant. Fortunately, guidelines exist for what is, in general, considered good acoustics. The physical basis for acoustics is given from section 2.2 to 2.3, the architectural view is found from section 2.5 to 2.6, the acoustical metrics commonly used, are described in section 2.7, whereas and the measuring techniques for determining the acoustic properties are introduced in section 2.8. To describe the performance of concert halls and auditoriums, the metrics: room gain (G), reverberation time (RT60), early decay time (EDT), speech intelligibility (C50), definition (D50), musical clarity, and temporal distribution, were conceived.

2.2 Introduction to sound

To establish sound as an entity, a lot of equations will be instantiated, with most coming from [7]. As most of the notation and equations in [7] are equal or at the very least similar to other sources, it has been chosen as the main source for equations, meaning all formulas used in this section can be found in [7] as well, unless otherwise noted. Other sources used for mathematical formulas are [4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. For the time being, only ideal sound will be described, assuming no losses in propagation or obstacles. Furthermore, the medium is considered homogeneous and at rest, making the velocity of sound constant in relation to space and time. If the medium were air, the velocity is given by:

$$c = 331.4 + 0.6 * T \left[\frac{m}{s}\right] \tag{2.1}$$

With T being the temperature in degrees centigrade. The speed of sound c is set to be $343\frac{m}{s}$ for all following computations, based on an average temperature of 23.2°. The average is used as exact calculations would require infinitesimal precision and, in practice, would be impossible to compute. An example could be a sports hall, wherein the temperature around players and audience is higher than under the rafters, [4, 5, 7, 10].

Another fundamental equation is the wave equation. The wave equation describes how sound propagates over time and space relative to the speed of sound. It is given by:

$$c^2 \Delta p = \frac{\partial^2 p}{\partial t^2} \left[\frac{Pa}{m^2} \right] \tag{2.2}$$

where the squared speed of sound is defined by:

$$c^2 = \kappa \frac{p_0}{\rho_0} \left[\frac{m^2}{s^2} \right] \tag{2.3}$$

in these equations, p is the sound pressure, rho_0 is the static gas density value ($\approx 1.2[\frac{kg}{m^3}]$, p_0 is the static sound pressure, and κ is the adiabatic component (for air $\kappa = 1.4$), [7]. In Cartesian coordinates, the Laplacian operator Δ is given by:

$$\Delta p = \frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2} \tag{2.4}$$

To elaborate, the wave equation is derived from the conservation of momentum relation:

$$\nabla p = -\rho_0 * \frac{\partial v}{\partial t} \tag{2.5}$$

with its one-dimensional version being:

$$\frac{\partial p}{\partial x} = -\rho_0 \frac{\partial v_x}{\partial t} \tag{2.6}$$

where: v is a vector representing particle velocity, and t is time. Furthermore, the mass conservation requirement leads to:

$$\rho_0 \; div \; v = -\frac{\partial \rho}{\partial t} \tag{2.7}$$

with ρ being the variable gas density. It is generally assumed that the variation in gas pressure and density is small compared to their static values. All of the above can be related by:

$$\frac{p}{p_0} = \kappa \frac{\rho}{\rho_0} = \frac{\kappa}{\kappa - 1} * \frac{\delta * T}{T + 273} \tag{2.8}$$

By eliminating the particle velocity v and variable gas density ρ from equations 2.5-2.8 one would arrive at the wave equation given in equation 2.2. The wave equation holds true for sound waves in any lossless fluid and for the resulting pressure, density, and temperature variations.

2.2.1 Actual waves

In sections "Plane Waves" and "Spherical Waves", ideal scenarios are described, and a more realistic approach is made in the following. Therefore, directional sound radiation with angular dependence is now considered. It is best imagined by a singular speaker pointing into an ideal environment. As the speaker is pointing in a direction and is not omnidirectional, the resulting sound wave will not be uniformly spherical, [7, 8, 21]. This expands the spherical wave equation to now include the polar and azimuth angles as well:

$$\frac{\partial^2 p}{\partial r^2} + \frac{2}{r} * \frac{\partial p}{\partial r} + \frac{1}{r^2} \left(\frac{1}{sin(\theta)} * \frac{\partial}{\partial * \theta} \left(sin(\theta) * \frac{\partial}{\partial * \theta} \right) + \frac{1}{sin^2(\theta)} * \frac{\partial^2}{\partial \phi^2} \right) = \frac{1}{c^2} * \frac{\partial^2 p}{\partial t^2} \tag{2.9}$$

To solve the complex spherical wave equation, the sound pressure can be expressed as a Fourier series separated into its radial and angular variables:

$$p(r,\theta,\phi) = A * \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \Gamma_{nm} h_n(kr) Y_n^m(\theta,\phi) \tag{2.10}$$

Where: Γ_{nm} are the Fourier series coefficients of order n and degree m, $h_n(kr)$ is the Hankel function of the first kind with order n, $Y_n^m(\theta,\phi)$ are the spherical harmonics, forming an orthonormal basis, which is used in the Legendre polynomial:

$$Y_n^m(\theta,\phi) = \sqrt{\frac{(2n+1)}{4*\pi} * \frac{(n-|m|)!}{(n+|m|)!}} * P_n^{|m|} * (cos(\theta) * \begin{cases} cos|m|\phi & if \;\; m \geq 0 \\ sin|m|\phi & if \;\; m < 0 \end{cases} \tag{2.11}$$

To actually use the above equations for anything, a spherical microphone array surrounding the speaker must be used to obtain empirical data. When such a measuring system is used, the coefficients of Γ_{nm} can be found with an inverse two-dimensional Fourier transform — often referred to as the inverse spherical harmonic transform.

It should be noted that degrees (n) and orders (m) of a spherical harmonic refer to the number of harmonics present along the radial and azimuthal angles. The degree n characterizes the overall angular frequency or spatial resolution of the wave pattern on the sphere, while the order m details the specific azimuthal variation, defining the symmetry of the sound field with respect to the polar axis.

2.2.2 Human Hearing and Perception

As 13 orders of magnitude differentiate the hearing threshold and the threshold for painful listening in the most sensitive part of human hearing (1-3kHz), it would not make sense to discuss the sound pressure, [7, 8, 12, 22, 23]. Therefore, sound pressure level SPL is used instead, given by:

$$SPL = 20 * log10\left(\frac{\tilde{p}}{\tilde{p_0}}\right)[dB] \tag{2.12}$$

Where $\tilde{p}$ denotes root mean square of the sound pressure, with $\sqrt{\tilde{p^2}}$ and $\tilde{p_0} = 2 * 10^{-5}[\frac{N}{m^2}]$ as an internationally fixed value corresponding to the hearing threshold at 1kHz, [7]. The sound pressure level reveals the objective sound pressure but tells nothing about the subjective sound level. A 1kHz sinusoid at 120dB is deafening, while a 30kHz sinusoid at the same SPL is inaudible. To compensate for this subjective perception, the unit "phon" and the associated "sone" are used. Phons are in an idealized world equal to the SPL of a 1kHz sinusoid at any given dB, and can therefore be approximated as $10phons = 10dB@1kHz$. To yield a better understanding of phons and sones, figure 2.1 shows what is known as equal loudness curves (also known as Fletcher-Munson curves) from the DS/ISO standard 226:2023, [24]. Equal loudness curves show the subjective loudness perception of pure-tone signals at varying frequencies with known SPL, and can be used to find the exact perception of loudness from 20Hz to 12500Hz. The threshold of hearing is found at the dashed line at the bottom, signifying 0 phon, established in DS/ISO 389-7, [25]. The curves for 10 and 100 phon are dotted lines as very little experimental data exists for these levels, [24]. *NOTE: the equal loudness curves are only applicable for pure-tone signals, as complex sounds, i.e., a hammer hitting a nail, contain*

masking of some spectral quantities, and can therefore not be placed specifically on the curves. In such an instance, dB(A) measurements should be used instead, [7, 24, 25].

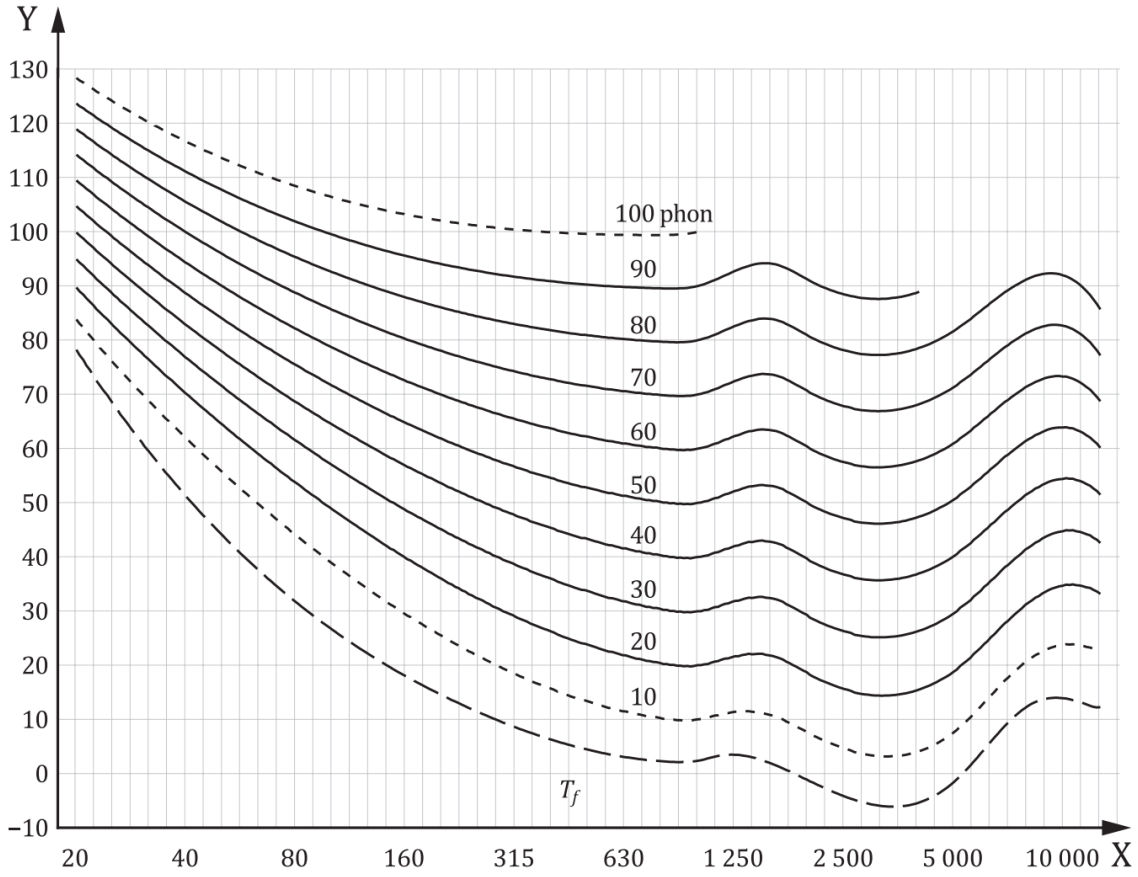


Figure 2.1: Equal loudness curves for pure tone listening in free field conditions, [24]. X is frequency, expressed in Hz, Y sound pressure level, expressed in dB, and T_f hearing threshold.

The metric "sone" describes when the subjective loudness is doubled, as doubling the number of phons does not equal a doubling in perceived loudness. The first sone equals 40 phons, and thereafter 1 sone equals 10 phons. The relation is best understood by viewing table 2.1, and is otherwise given by equations 2.13 and 2.14, with N being loudness in sones and L_N being loudness in phons.

$$N = \left(10^{\frac{L_N - 40}{10}}\right)^{0.30103} \approx 2^{\frac{L_N - 40}{10}} \tag{2.13}$$

$$L_N = 40 + log_2(N) \tag{2.14}$$

Phon	0	10	20	30	40	50	60	70	80	90	100
Sone	0 (Inaudible)	0.019	0.138	0.439	1	2	4	8	16	32	64

Table 2.1: Relationship between phons and sones, calculated using the Zwicker method in Matlab, as described in ISO 532-1, [26, 27].

The remaining aspects related to human perception of sounds are mostly oriented towards psychoacoustics and binaural perception, such as head-related transfer functions (HRTF) and binaural transfer functions,[7, 28, 29]. While the HRTF is applicable

in the virtual auralization of rooms, it will not be expanded upon in this project, and neither will psychoacoustical properties in depth.

2.3 Sound in Front of a Wall

Instead of ideal free-field propagation, actual constraints such as a wall will now be introduced. A wall will have some inherent properties, such as its reflection, diffusion, transmission, resonance, and absorption properties, that together determine the actual acoustical properties of each specific wall. In table 2.2, each property and its typical values are presented, with relevant equations following.

Property	**Definition**	**Typical Values**
Reflection Coefficient (R)	Fraction of sound energy reflected	0.8 - 0.98 (Concrete), 0.1 - 0.5 (Foam)
Diffusion Coefficient (D)	Measure of even sound scattering	0.2 (Flat wall), 0.6 - 0.9 (Diffuser)
Transmission Loss (TL)	Reduction of sound through a barrier (dB)	25 dB (Drywall), 50+ dB (Concrete)
Resonance Frequency (f_r)	Frequency at which the wall vibrates	50 - 500 Hz (Depending on material)
Absorption Coefficient (α)	Fraction of energy absorbed	0.02 (Concrete), 0.8+ (Acoustic panels)
Wall Impedance (Z)	Opposition to sound transmission, depends on density and stiffness.	1.5×10^6 kg·m^{-2}·s^{-1} (Concrete), 5×10^5 kg·m^{-2}·s^{-1} (Brick), 10^4 kg·m^{-2}·s^{-1} (Foam)
Wall Admittance (ζ)	Measure of a wall's ability to accept sound energy, reciprocal of impedance.	0.001 (Concrete), 0.003 (Brick), 0.1 (Foam)

Table 2.2: Acoustic properties of walls and their typical values, [30, 16, 15].

- **Reflection Coefficient:**

$$R = \frac{I_r}{I_i} \tag{2.15}$$

where: I_r is reflected intensity and I_i is incident intensity.

- **Diffusion Coefficient:**

$$D = 1 - \frac{\sum S(\theta)}{N S_{\text{ideal}}} \tag{2.16}$$

where: $S(\theta)$ is the scattered energy in direction θ, and S_{ideal} is the ideally diffused scattered energy.

- **Transmission Loss (TL):**

$$TL = 10 \log_{10} \left(\frac{I_i}{I_t} \right) \tag{2.17}$$

where: I_t is the transmitted intensity.

- **Resonance Frequency (Panel):**

$$f_r = \frac{60}{d} \sqrt{\frac{E}{\rho(1-\nu^2)}} \tag{2.18}$$

where: d = panel thickness, E = Young's modulus, ρ = density, and ν = Poisson's ratio.

- **Absorption Coefficient:**

$$\alpha = 1 - |R|^2 \left[\frac{sabin}{m^2} \right] \tag{2.19}$$

- **Wall Impedance:**

$$Z = \left(\frac{p}{v_n} \right) \tag{2.20}$$

where: v_n denotes the particle component normal to the wall.

- **Wall Admittance:**

$$\zeta = \frac{Z}{\rho_0 * c} \tag{2.21}$$

All walls are assumed to be plane, unbounded, and smooth. Moreover, the sound source will be placed at a distance allowing the spherical waves to be considered planar.

2.3.1 Reflection at Normal Incidence

When referring to normal incidence, a reflection perpendicular to the wall should be imagined. To comply with previous formulas, the axis of propagation remains along the x-axis, resulting in a wall being aligned with the z-axis and the y-axis. Furthermore, the wall is set to $x = 0$ and the wave will arrive from the negative direction, yielding the following sound pressure:

$$p_i(x,t) = \hat{p}_0 * e^{i*(\omega*t-k*x)} \tag{2.22}$$

And particle velocity:

$$v_i(x,t) = \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t-k*x)} \tag{2.23}$$

Now that the sound wave has hit the wall, the reflected wave in completion is given by the wall's characteristics. For now, only the reflection coefficient R is used, as that can help describe both the amplitude attenuation and the phase shift, which also results in k flipping signs. Following equation C.5, the sign should be flipped for the reflected particle velocity as well. By using this, the reflected waves' sound pressure and particle velocity are given by:

$$p_r(x,t) = R * \hat{p}_0 * e^{i*(\omega*t+k*x)} \tag{2.24}$$

$$v_r(x,t) = -R * \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t+k*x)} \tag{2.25}$$

To find the impedance of the wall, x should be set equal to the coordinates of the wall (0), yielding equations 2.26 and 2.27. It should be noted in those equations that they are a product of both the incident and reflected sound wave, yielding the $1 + R$ and $1 - R$ relations:

$$p(0,t) = (1+R) * \hat{p}_0 * e^{i*(\omega*t)} \tag{2.26}$$

$$v(0,t) = (1-R) * \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t)} \tag{2.27}$$

With the sound wave evaluated at the wall, it is possible to use equation 2.20 to derive the exact wall impedance:

$$Z = \left(\frac{p}{v_n}\right) \rightarrow Z = \rho_0 * c * \frac{1+R}{1-R} \tag{2.28}$$

Which can be reformulated for the wall admittance, given in equation 2.21:

$$\zeta = \frac{Z}{\rho_0 * c} \rightarrow \zeta = \frac{\rho_0 * c * \left(\frac{1+R}{1-R}\right)}{\rho_0 * c} \rightarrow \zeta = \frac{1+R}{1-R} \tag{2.29}$$

With this, the reflection coefficient can also be described as:

$$R = \frac{\zeta - 1}{\zeta + 1} \tag{2.30}$$

And the absorption coefficient can be found as:

$$\alpha = \frac{4 * Re(\zeta)}{|\zeta|^2 + 2 * Re(\zeta) + 1} \left[\frac{sabin}{m^2}\right] \tag{2.31}$$

Graphically the wall impedance can be seen in figure 2.2. As α increases, the circles draw nearer to $\zeta = 1$ corresponding to impedance matching of the medium. A completely rigid wall will have infinite impedance, as $R = 1$, while a soft wall with $R = -1$ will yield no impedance.

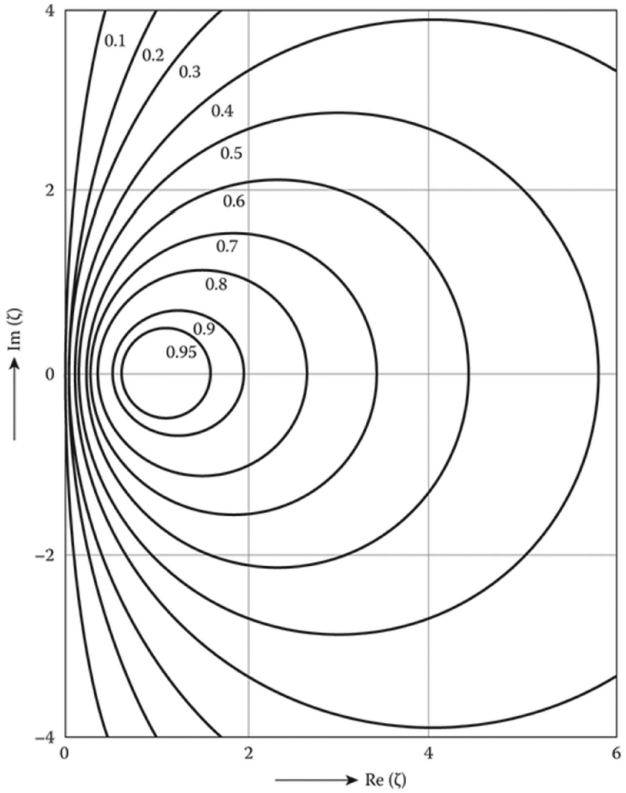


Figure 2.2: Circles of constant absorption coefficient in the complex wall impedance plane for normal sound incidence, with the absorption coefficient values indicated by the numbers next to the circles, [8].

2.3.2 Reflection at Oblique Incidence

In reality, some reflections happen at normal incidence, but due to the spherical propagation of sound, a significantly larger amount will have its incidence at oblique angles. Therefore, θ will be added to all previous calculations, representing the angle between the incident wave and the normal to the surface. By doing so, the coordinate system is transformed with $x \rightarrow x'$, defining a new propagation axis, which is given by:

$$x' = x * cos(\theta) + y * sin(\theta) \tag{2.32}$$

With x' the coordinate system expands from only regarding the x-axis to being two-dimensional, including the y-axis. Modifying the calculations from normal incidence to oblique incidence is done simply by replacing x with x'. This yields modified expressions for the incident and reflected wave properties. The key equations are:

- **Incident sound pressure:** Equation (2.33)

$$p_i(x',t) = \hat{p}_0 * e^{i*(\omega*t-k*(x*cos(\theta)+y*sin(\theta)))} \tag{2.33}$$

- **Incident particle velocity:** Equation (2.34)

$$v_i(x',t) = \frac{\hat{p_0}}{\rho_0 * c} * cos(\theta) * e^{i*(\omega*t-k*(x*cos(\theta)+y*sin(\theta)))} \tag{2.34}$$

- **Reflected sound pressure:** Equation (2.35)

$$p_r(x',t) = R * \hat{p}_0 * e^{i*(\omega*t+k*(x*cos(\theta)+y*sin(\theta)))} \tag{2.35}$$

- **Reflected particle velocity:** Equation (2.36)

$$v_r(x',t) = -R * \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t+k*(x*cos(\theta)+y*sin(\theta)))} \tag{2.36}$$

- **Wall impedance:** Equation (2.37)

$$Z = \left(\frac{p}{v_n}\right) \rightarrow Z = \frac{\rho_0 * c}{cos(\theta)} * \frac{1+R}{1-R} \tag{2.37}$$

- **Reflection coefficient:** Equation (2.38)

$$R = \frac{Z * cos(\theta) - \rho_0 * c}{Z * cos(\theta) + \rho_0 * c} = \frac{\zeta * cos(\theta) - 1}{\zeta * cos(\theta) + 1} \tag{2.38}$$

- **Absorption coefficient:** Equation (2.39)

$$\alpha(\theta) = \frac{4 * Re(\zeta) * cos(\theta)}{(|\zeta| * cos(\theta))^2 + 2 * Re(\zeta) * cos(\theta) + 1} \tag{2.39}$$

- **Sound pressure at the wall:** Equation (2.40)

$$p(x,y) = \hat{p}[1 + |R|^2 + 2 * |R| * cos(2 * k * x * cos(\theta) + \chi]^{1/2} * e^{-i*k*y*(sin\theta)} \tag{2.40}$$

- **Particle velocity at the wall:** Equation (2.41)

$$c_y = \frac{\omega}{k_y} = \frac{\omega}{k * sin(\theta)} = \frac{c}{sin(\theta)} \tag{2.41}$$

As with the reflections at normal incidence, the reflected wave will have a phase shift and attenuation, which leads to signs being flipped when looking at the reflected wave. Furthermore, the impedance, reflection coefficient, and absorption coefficients are evaluated at $x = 0$, corresponding to the location of the wall.

2.4 Closed Space Sound Field

With reflections for normal and oblique incidents described, sound propagation in closed spaces can be described. In closed spaces, the propagation is notably different from free-field due to the reflections interacting with the sound source, other reflections, and boundaries anew, creating even more reflections, [7, 8, 31, 10, 11, 5, 18]. This continuous interaction between reflected waves and sound sources creates modal interactions, reverberation, and standing waves, which all are determining factors in a room's acoustics. To create a representation of the associated wave theory, the wave equation 2.2 is modified to resemble the Helmholtz equation:

$$\Delta p + k^2 p = 0 \tag{2.42}$$

The Helmholtz equation describes wave behavior and sound propagation in closed spaces, by the spatial variation of sound pressure fields with harmonic time dependencies. Therefore, it can also only be used to describe waves under the assumption of time harmonic waves with an angular frequency ω. This is given by:

$$\frac{\partial^2}{\partial t^2}\left(P(x)e^{i\omega t}\right) = -\omega^2 P(x)e^{i\omega t} \rightarrow -\omega^2 P(x)e^{i\omega t} = c^2\nabla^2 P(x)e^{i\omega t} \rightarrow \Delta p + k^2 p = 0 \tag{2.43}$$

Where the second time derivative is taken and substituted into the wave equation. Thereafter $e^{i*\omega*t}$ is canceled as it is on both sides of the equation, minimizing the expression to equal 2.42. The wave number k is an eigenvalue, as the wave equation can only yield non-zero solutions for equations 2.44 and 2.45 for particular discrete values, as k is still given by $k = \frac{\omega}{c}$. For k values at specific nodes, equation 2.46 is used.

$$Z * \frac{\partial p}{\partial n} + i * \omega * \rho_0 * p = 0 \tag{2.44}$$

$$\zeta * \frac{\partial p}{\partial n} + i * k * p = 0 \tag{2.45}$$

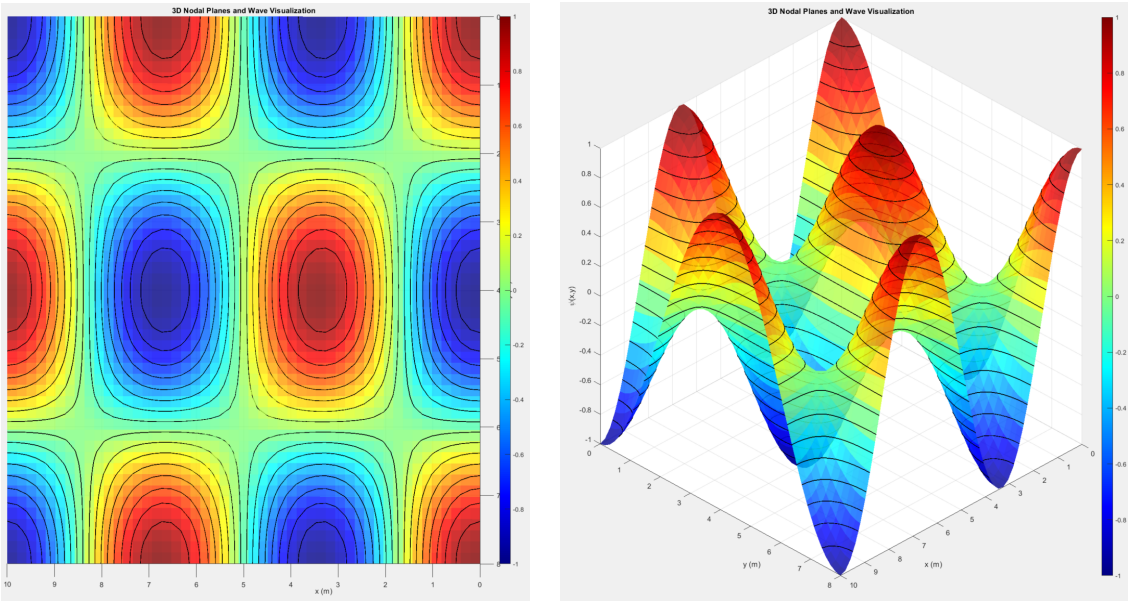
$$k_{n_x n_y n_z} = \pi * \left(\left(\frac{n_x}{L_x}\right)^2 \left(\frac{n_y}{L_y}\right)^2 \left(\frac{n_z}{L_z}\right)^2\right)^{\frac{1}{2}} \tag{2.46}$$

In a rectangular room of dimensions, $L_x \times L_y \times L_z$, the general solution for the pressure distribution considering rigid boundaries is given by:

$$p(x,y,z,t) = P_0 \cos\left(\frac{n_x \pi x}{L_x}\right) \cos\left(\frac{n_y \pi y}{L_y}\right) \cos\left(\frac{n_z \pi z}{L_z}\right) e^{i\omega t}, \tag{2.47}$$

Where n_x, n_y, n_z are index numbers along the respective axes, signifying which nodal plane is being referred to. Equation 2.47 describes a three-dimensional standing wave, wherein the sound pressure is zero for all times, at any point where the cosines equal to zero. All values of x (and y or z) which are odd integers of $\frac{L_x}{n_x}$ result in cosines equal to zero, [7, 8]. The intersection between the x, y, and z-axis equidistant planes is referred to as nodal planes, which are mutually orthogonal. It should be noted that nodal surfaces differ from nodal planes, as nodal surfaces are used to describe non-orthogonal surfaces, which may not even be planar. The intersections between nodal planes will therefore also always have a sound pressure of zero, [7, 8]. In figure 2.3 a sound pressure distribution can be seen in an 8x10 meter room subjected to 40 and

50Hz. In other words, this particular room has 3 room nodes in the x direction and 2 room nodes in the y direction, at z=0.



(a) Nodal planes seen from the top.

(b) Nodal planes as seen from a corner.

Figure 2.3: Nodal plane illustrations in an 8x10 meter room subjected to 40 and 50Hz.

By using equations 2.47 eigenvalues can be found, and with those in place, the eigenfrequencies of a room can be found with equation 2.48, where $k_{n_x n_y n_z}$ is given by equation 2.46. In table 2.3, the first 20 eigenfrequencies of a room with dimensions 4.7 × 4.1 × 3.1 m can be seen, using 340 m/s as c.

$$f_{n_c n_y n_z} = \frac{c}{2 * \pi} * k_{n_x n_y n_z} \tag{2.48}$$

fn	nx	ny	nz	fn	nx	ny	nz
36.17	1	0	0	90.47	1	2	0
41.46	0	1	0	90.78	2	0	1
54.84	0	0	1	99.42	0	2	1
55.02	1	1	0	99.80	2	1	1
65.69	1	0	1	105.79	1	2	1
68.75	0	1	1	108.51	3	0	0
72.34	2	0	0	109.68	0	0	2
77.68	1	1	1	110.05	2	2	0
82.93	0	2	0	115.49	1	0	2
83.38	2	1	0	116.16	3	1	0

Table 2.3: Eigenfrequencies of a rectangular room with dimensions 4.7 × 4.1 × 3.1 m (in Hz), [7].

To find the number of room modes in a room, every combination of indexes must be calculated for a given frequency range. This is done by equations 2.49 and 2.50 and is realistically done by looping a code such as ***HER SKAL INDSÆTTES KODELINK***, as the amount of computations necessary is unfathomably large. Using the same room as in [7], a frequency threshold of 200 Hz has 78 modes and 71 unique eigenfrequencies,

whereas a threshold of 2 kHz has a threshold of 52.126 modes and 9.850 eigenfrequencies. If the hearing spectrum up to 20 kHz is used, it leads to a staggering 49.855.567 modes and 143.378 unique eigenfrequencies.

$$\left(\frac{2 * f_{max}}{c}\right)^2 = \left(\frac{2 * 20000}{c}\right)^2 = 116.619^2 \tag{2.49}$$

$$\left(\frac{n_x}{L_x}\right)^2 + \left(\frac{n_y}{L_y}\right)^2 + \left(\frac{n_z}{L_z}\right)^2 \leq 116.619^2 \tag{2.50}$$

2.5 Room Geometry in Relation to Acoustics

Until now, only rectangular rooms have been discussed, as they form a more intuitive understanding. When more flamboyant room geometry is in use, all sound propagation is thought of as rays instead of spheres. By using sound rays to describe propagation instead of spheres, simplifies the understanding of a room's reflections by reducing the computational complexity and increasing the graphical understanding, [7, 8, 10, 11, 15, 16, 31]. It is possible by the limiting case of very high frequencies, where any frequency above 1 kHz is considered high, as the wavelength (34.3cm) is considered small in comparison with typical room dimensions.

2.5.1 Image Sources

Image sources are an intuitive (yet conceptual) way to describe reflections using sound rays. Figure 2.4 shows a basic image source at A'. Image sources are best described as an identical sound source placed mirrored according to a wall. The original sound source A has in figure 2.4 3 sound rays hitting a wall. As the wall is considered smooth, the reflection will have an identical angle of departure as the angle of incidence, [7, 8, 10, 11, 15, 16, 31]. This is furthermore underlined as identical sound rays propagate from A'.

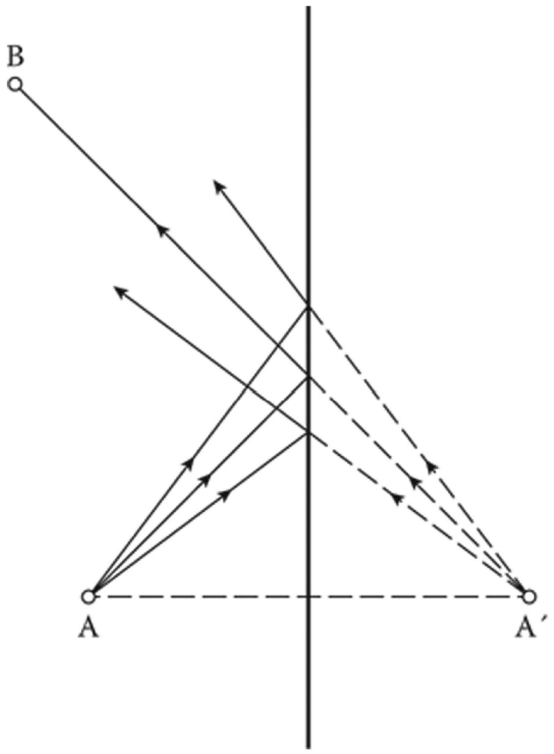


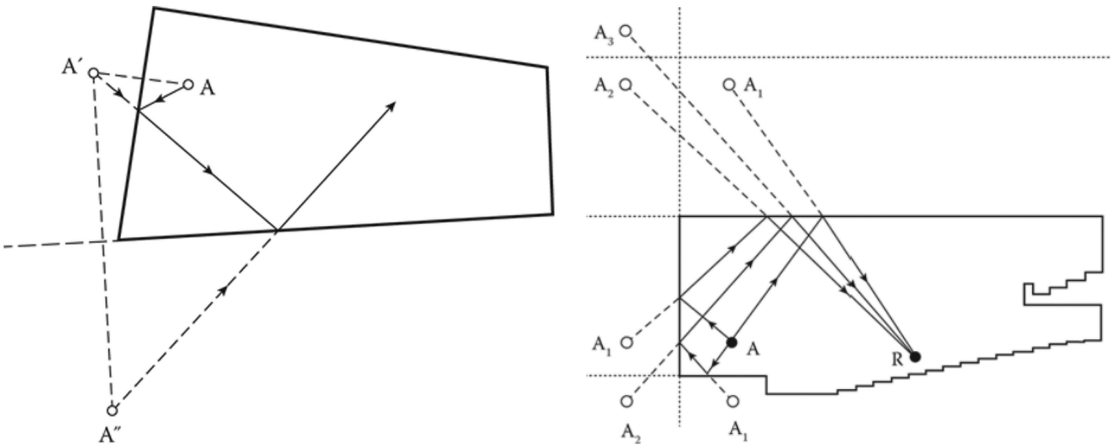
Figure 2.4: Original sound source A and image source A', [7]. Note the angle of incidence and departure for sound rays.

In this exact example, the image source is meant to determine where on the wall A's reflection will coincide with B, which is also why the direct sound ray is neglected. The image source finds that the middle sound ray is the only matching reflection, using sound rays at least. It should be remembered that sound sources still propagate omnidirectionally unless otherwise specified. It should also be noted that for now, reflection/absorption coefficients are currently not used to describe ray propagation. If those coefficients were applied, the sound rays would decay accordingly, [7, 8,

10, 11, 15, 16, 31].

Figure 2.5 is a more interesting example, as a non-rectangular room is used. The first image source A' is placed mirrored to the original source, but the second image source A'' is not mirrored to the original source. Instead, the image source A'' is the image source to A' if the mirror plane is placed at the next wall, which the reflection should hit. This can be done continuously unless reflection/absorption coefficients are used.

In general, image sources and their corresponding reflection diagrams are used with 3-5 orders for initial investigation, and by using programs such as ODEON, higher-order diagrams can be made.



(a) First and second order reflections in an oddly shaped room, [8].

(b) Image sources in an auditorium, found by mirroring along the walls being hit, [8].

Figure 2.5: In an oddly shaped room, image sources are best found by mirroring the source along the wall, which the reflection hits next. A', A_1, etc denotes the order of image sources, [8].

A significant detail regarding image sources is that they are infinite in the same way as orthogonal mirrors create an infinite space. As most rooms have at least four walls, a floor, and a ceiling, image sources will multiply incredibly fast. For a room with N plane walls, the total number of image sources for a given order i is $N * (N-1)^{i-1}$, for $i \geq 1$. The total number of images is given by equation 2.51, which for $N = 4$ and $i = 5$ yields a total of 160 image sources:

$$N(i_0) = N * \frac{(N-1)^{i_0} - 1}{N-2} \tag{2.51}$$

Moreover, as ceilings and floors has to be taken into consideration as well, an amount of mirrored rooms will also be present above and below the original room, each with similar amounts of image sources for their respective plane. For a simple rectangular room, this could look as shown in figure 2.6, where the image should be imagined as one plane, with a mirrored one on top and below for order i.

Image sources/image rooms can be used to determine the rate of reflections, yielding a temporal distribution, [7, 8, 10, 11, 15, 16, 31]. The temporal distribution describes the intensity and time dependency of reflections. Using equation 2.52, the temporal distribution can be calculated for each time t if the absorption coefficients of all walls were frequency independent. However, as all materials are frequency dependent, equation 2.53 is used instead, as it describes the response of a single pulse, with A_n being the strength and t_n the time of arrival for reflection n. The temporal distribution is

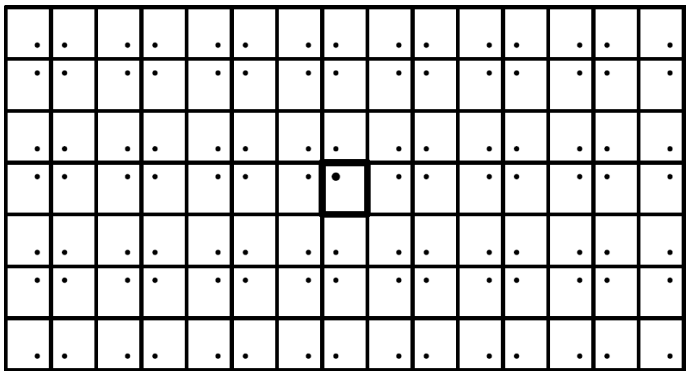


Figure 2.6: Image sources as they appear in adjacent theoretical rooms, with the original room in the middle. This plane of image sources is multiplied by the order i of image planes.

partitioned into: direct sound, early reflections, and reverberation. The time delay and intensity of the early reflections, as well as the reverberation, are detrimental to the reverberation of a room, and in general for a room's acoustical properties, as described in section 2.7.2 on page 21. In figure 2.7, a temporal distribution can be seen using an echogram. The echogram shows the pulses and their intensity along the time axis, with the first pulse being the direct sound at $t = 0$. All subsequent reflections carry less and less density, but will also increase exponentially in density, as an increasing amount of continuously weaker reflections appear.

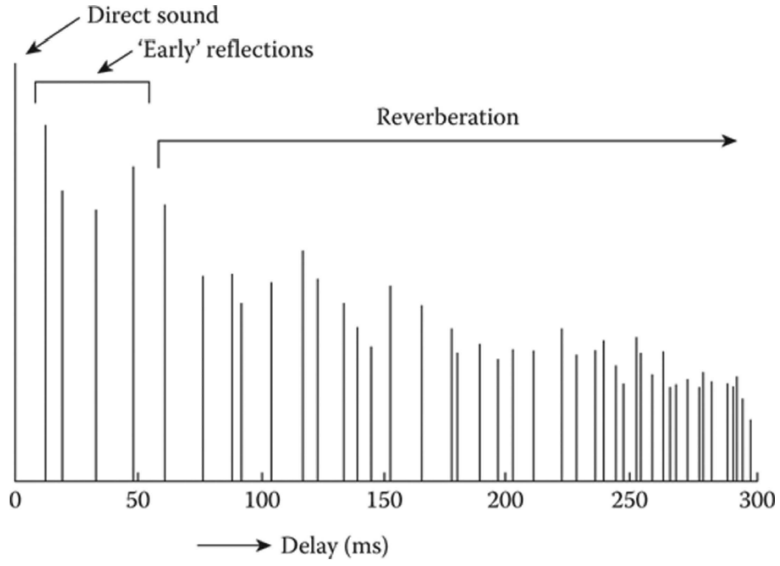


Figure 2.7: Temporal distribution showing the impulse response of a room, [8].

$$s'(t) = \sum_n A_n * s(t - t_n) \tag{2.52}$$

$$g(t) = \sum_n A_n * \delta(t - t_n) \tag{2.53}$$

2.6 Sound Absorbers

Sound absorbers are present in all acoustical measurements, no matter the material. By absorbers, an attenuation of the original sound pulse is meant. That attenuation can either be achieved by conversion into heat, transmission to a different medium, or simply by propagation in free space, which essentially is a conversion into heat as well. In air, the attenuation coefficient m is found by dividing the absorption coefficient by the distance in meters:

$$m = \frac{\alpha}{distance} \tag{2.54}$$

Or from a known intensity it can be found by:

$$I(x) = I_0 * e^{-m*x} \tag{2.55}$$

In table 2.4, the attenuation relative to humidity and frequency can be seen. Note that with an increase in frequency, the attenuation increases as well. This is due to the non-ideality of the real world, as the mechanical energy irreversibly converts into heat, as sound waves propagate through air. Furthermore, sound waves create an oscillating rarefication and compression of air molecules, which mechanically will create friction between them, thus creating heat.

	Frequency (kHz)						
Relative Humidity (%)	0.5	1	2	3	4	6	8
40	0.60	1.07	2.58	5.03	8.40	17.71	30.00
50	0.63	1.08	2.28	4.20	6.84	14.26	24.29
60	0.64	1.11	2.14	3.72	5.91	12.08	20.52
70	0.64	1.15	2.08	3.45	5.32	10.62	17.91

Table 2.4: Attenuation constant m of air at 20°C and normal atmospheric pressure, in 10^{-3} m^{-1}, [7, 8].

Apart from heat, sound can also be "absorbed" by transmission, such as with open windows and doors. A more complex scenario is the transmission of sound through a wall or other structural material. A perfectly rigid and reflecting wall does not exist, therefore, we assume that some value different from zero will always be absorbed or transmitted through any wall. If the sound pressures of an incident sound wave are denoted p_1, p_2, and p_3 for the incident, reflected, and transmitted components, the pressure acting on a wall can be described by $p_1 + p_2 - p_3$. The inertial force of the wall balances this by $i * \omega * M * v$, where M is the mass per unit area and v is the velocity of motion of the wall. This is equal to the particle velocity of the wave radiating from the back of the wall, given as $p_3 = \rho_0 * c * v$. From this, the impedance of a wall can be given as:

$$Z = \frac{p_1 + p_2}{v} = i * \omega * M * v + \rho_0 * c \tag{2.56}$$

Based upon equation 2.20. Implementing the wall admittance given by equation 2.21 as well, yields the following absorption coefficient, also known as the mass law:

$$\alpha = \left(\frac{2 * \rho_0 * c}{\omega * M} \right)^2 \tag{2.57}$$

The above calculations are only available for walls whose mass reactance is large in comparison with the characteristic impedance of air. As frequency decreases, the

absorption coefficient drops as well, and poses the fact that lower frequencies will transmit from one room to the next, more easily than high frequencies, due to the impedance mismatch between air and the wall. Typically, this is dealt with by impedance matching or by increasing the mass of the wall per unit area, if lower transmission is desired, [7, 8, 10, 11, 15, 16, 31, 32, 33, 34, 35]. If higher absorption is desired, lighter materials such as foam or similar porous materials should be applied. A result of lower absorption coefficients for lower frequencies is that a room can sound more "crisp", as the sound wave is neither absorbed nor reflected, but rather transmitted by the wall, whereas higher frequency sound waves yield a larger absorption and reflection coefficient.

2.6.1 Absorbers in Construction

In construction, absorbers are present in many forms, such as perforated walls, resonance absorbers, porous materials, heavy walls, and acoustically optimized windows. Perforated walls are commonly found in acoustic panels or as stand-alone walls placed some distance from a rigid backwall. The perforated plates act as both a reflecting medium and a transmitting medium, as each hole can be considered a short tube with length b equal to the thickness of the material, , [7, 8, 10, 11, 15, 16, 31, 5, 17]. The air within each hole is given by $\rho_0 * b$. A perforated panel has an "equivalent mass", which is given by equation 2.58, where σ is given by the perforation ratio $\frac{S_1}{S_2}$, wherein S_1 is the area of the whole and S_2 is the panel area per hole. The remaining variable b' is given by the effective length $b' = b + 2 * \delta b$, wherein δb is the end correction of a tube, which considers the fact that pressure waves cannot contract and expand abruptly when entering or leaving an aperture. For a circular hole with radius a, the end correction is given by $\delta b = \frac{\pi}{4} * a$. For a rectangular hole, the end correction is approximated by $\delta b = \frac{\pi}{4} * \frac{a+b}{2}$. With these variables in place, it is possible to calculate the absorption coefficient of the perforated panel, as shown in equation 2.59, where M' is the equivalent mass and M'_0 is the specific mass for all solid parts of the panel, [7, 8].

$$M' = \frac{\rho_0 * b'}{\sigma} \tag{2.58}$$

$$M = \left(\frac{1}{M'} + \frac{1}{M'_0} \right)^{-1} \tag{2.59}$$

By calculating all of the above, the absorption coefficient can be found by updating variables with their respective values and calculating equation 2.57 with 2.59 as M.

Another practical absorption method that can be implemented in the construction phase is the integration of resonance absorbers. The general idea is to place a membrane in front of a cavity, where the membrane will be excited by a sound wave hitting it, thus creating vibrations controlled by the specific mass M and the cavity behind it. These vibrations represent an absorption and can be represented with vibrational losses, represented by the loss resistance r_s. Using equation 2.31 and 2.60, where d is the distance between the rigid wall and the membrane and M' is the specific mass of the membrane, the absorption coefficient can be calculated for various ratios of $\frac{r_s}{\rho_0 * c}$ under the assumption that $M' * \omega = 10 * \rho_0 * c$.

$$Z = r_s + i \left(\omega * M' - \frac{\rho_0 * c^2}{\omega * d} \right) \tag{2.60}$$

As r_s gets closer to $\rho_0 * c$, the absorption coefficient increases, until $\alpha = 1$ is achieved by $r_s = \rho_0 * c$. For all values far from r_s, the ratio yields broadening curves, as can be seen in figure 2.8.

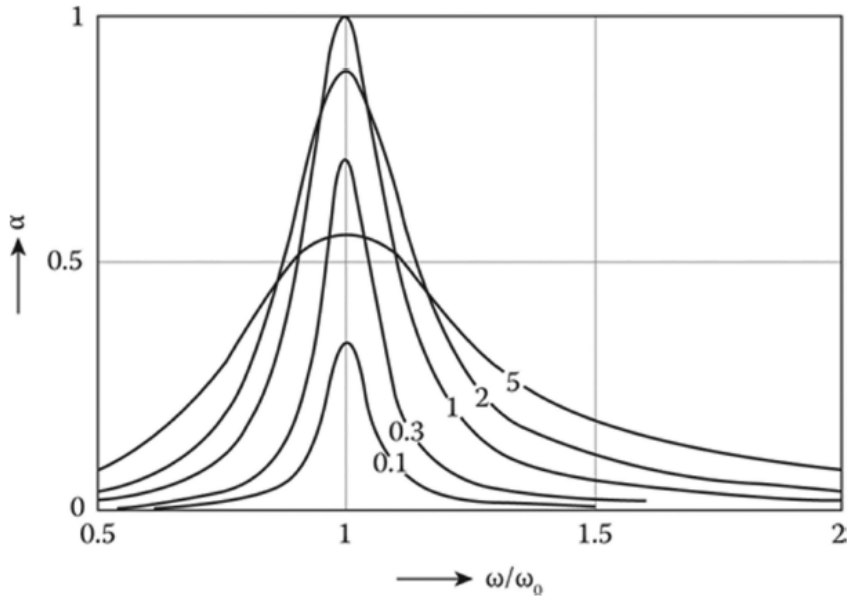


Figure 2.8: Calculated absorption coefficients for resonance absorbers at normal sound incidence, with determining parameter $r_s = \rho_0 * c$,[8].

When using resonance absorbers in construction, the membrane often consists of plywood, plasterboard, metal plates, or similar materials. Depending on the resonating medium, different hurdles can arise, such as intrinsic impedance, elastic deformation, or other properties inherent to the material. Furthermore, the material must be attached to the rigid backwall, requiring more substantial fastening for some materials than others. A method for adapting the materials' characteristics and thereby r_s, is filling a percentage of the cavity with porous material, such as rockwool or foam. For normal sound incidence, the practical resonance frequency of an absorber is given by equation 2.61:

$$f_0 = \frac{600}{\sqrt{M' * d}}[Hz] \tag{2.61}$$

Where M' is in $kg\ m^{-2}$ and d in cm. If the cavity is filled with porous material, the value "600" should be replaced by "500", to account for it, [7]. As this equation is a practical formula rather than a scientifically based formula, it can only give a ballpark for the resonant frequency, as the mounting method will have a significant impact on the actual resonant frequency. The only way to get better results would be to test the construction in a reverberation chamber, examining the absorption coefficient.

Porous materials such as Rockwool, Troldtekt, Rockfon, and similar acoustic-altering materials work by employing the principle of viscous and thermal processes at the boundary layer. The boundary layer is between 0.01 and 0.2mm, depending on frequency. For smooth surfaces the absorption is rather small, but as the surface gets rougher, the absorption increases. Therefore, as a porous material can be described as an almost infinitely rough surface, the absorption increases dramatically, with the best effect achieved by channels/openings to the outside air. As infinitely many surfaces appear, an equal number of reflections will appear as well. Depending on the frequency,

these reflections can interfere destructively or constructively. However, because of the infinite number of waves present within the porous layer of decreasing strength, the attenuation should increase in relation to the depth of the medium.

The pressure fluctuations within the porous material/along the edges of each surface will result in a considerable amount of mechanical energy being withdrawn from the impinging soundfield, thus converting it into heat. Due to the inherently rough surface of some construction materials, such as bricks, mortar, and plaster (*not plasterboard*) will show similar characteristics as the porous materials. Unfortunately, as the rough layer is not nearly deep enough before becoming solid, the acoustical benefits of these materials are not comparable to products designed for absorption.

The characteristics of porous materials can be described mathematically by e.g. the Rayleigh model. However, mathematically describing such materials is too comprehensive and time-consuming, compared with measuring them in a reverberation room, [7, 8]. Therefore, the mathematics behind it is left out of this project.

2.6.2 Absorbers in Furnishment

Furnishment is the go-to solution for altering the acoustic properties of a room if it cannot be done structurally. Some typical examples of furniture/decor that will alter the acoustical properties are:

- **Carpets and Rugs:**
  Thick soft carpets, large rugs, or similar material that can shield off some of the hard flooring would be able to reduce the reflection coefficient of the floor significantly. Due to the large area and hard backing, low-frequency attenuation is achievable with the correct composition.

- **Upholstered furniture:**
  Lounge chairs, sofas, regular chairs, pillows, and other furniture can beneficially be upholstered instead of leathered/wooden, to create a porous surface with foam, felt, or similar backing. With correct placement in a room, the furniture can help control reflections.

- **Curtains:** Curtains can lower the reverberation time significantly if made of sufficiently thick or heavy material. Not to say that thin drapes will not have an effect, as such could act as resonance absorbers. Furthermore, louvers can be fitted to direct reflections in specific directions.

- **Acoustic images, art, etc.:** The latest craze is adding acoustic images to a room to alter the reverberation time. These are often composed of printed fabric wrapped around a porous material. Another option is art sculpted into the porous material, enabling effective attenuation in specific frequency bands to be placed on walls, ceilings, or other flat surfaces.

While all of the above is applicable in most rooms, other situations require more substantial incorporation. One such situation is improving the acoustics of a concert hall, lecture room, auditorium, sports hall, cinema, or similar large-scale hall. As such rooms are meant to be used by a large number of people, the acoustic properties must be determined by including the number of people in the equation. The absorption coefficient of a guest in any room is highly reliant on the clothing, as different fabrics, styles, and seasonal clothing alter the final measurement. Apart from clothing, the arrangement of an audience is highly influential as well, as the density is determined by

the number of people per square meter. In table 2.5 the absorption coefficient of a person can be viewed, obtained by placing them in a reverberation room. Unfortunately, this method does not take into account what would happen if a large amount of people were placed closely together, as reverberation rooms large enough to handle several hundred people do not exist. An alternative is measuring the change of reverberation in finished halls, but since most halls vary in size and shape, the results are rarely applicable to other halls than the measured. It is however, still possible to get useful measurements in a reverberation room, which can be seen in table 2.6, where the absorption coefficients of different chairs with an audience are shown.

Kind of Person	**125 Hz**	**250 Hz**	**500 Hz**	**1,000 Hz**	**2,000 Hz**	**4,000 Hz**
Male standing in heavy coat	0.17	0.41	0.91	1.30	1.43	1.47
Male standing without coat	0.12	0.24	0.59	0.98	1.13	1.12
Musician seated, with instrument	0.60	0.95	1.06	1.08	1.08	1.08

Table 2.5: Equivalent absorption area of persons, in m², [8].

Type of Seats	**125 Hz**	**250 Hz**	**500 Hz**	**1,000 Hz**	**2,000 Hz**	**4,000 Hz**	**6,000 Hz**
Audience seated on wooden chairs, two persons per m²	0.24	0.40	0.78	0.98	0.96	0.87	0.80
Audience seated on wooden chairs, one person per m²	0.16	0.24	0.56	0.69	0.81	0.78	0.75
Audience seated on moderately upholstered chairs, 0.85 m × 0.63 m	0.72	0.82	0.91	0.93	0.94	0.87	0.77
Audience seated on moderately upholstered chairs, 0.90 m × 0.55 m	0.55	0.86	0.83	0.87	0.90	0.87	0.80
Moderately upholstered chairs, unoccupied, 0.90 m × 0.55 m	0.44	0.56	0.67	0.74	0.83	0.87	

Table 2.6: Absorption coefficients of audience and chairs (reverberation chamber), [8].

As can be seen from table 2.6, the seat type has a pronounced effect on absorption, especially at low frequencies, where upholstered seats more than triple the absorption coefficient, compared with wooden chairs. Unfortunately, as seating is divided into blocks divided by pathways rather than uniformly across a hall, exact absorption is best found by measuring the reverberation time before and after the installation of seats in a hall.

An effect that has been measured in finished concert halls is known as "the seat dip

effect". This effect is the attenuation present across empty rows of seats, with the highest attenuation from 80-250 Hz. While the effect has been replicable for occupied seats as well, it is not as common as for unoccupied seats, [7, 8, 10, 11].

The attenuation of direct sound is furthermore increased with an increased amount of rows in front of the listening position. One aspect is the natural attenuation in air, but the most prominent factor is diffraction of sound, as it hits seats and occupants of the rows in front. A typical countermeasure to this phenomenon is the vineyard topology of many concert halls. That is, the placement of seats on a slope maximizes the area that is subjected to direct sound while minimizing grazing incidence.

2.7 Metrics for Room Acoustics

2.7.1 Room Gain - G:

The gain of a room is very literally the amplification or attenuation that a room provides. It can be measured using a calibrated omnidirectional speaker (such as a dodecahedral speaker), where an impulse is first measured in a free field (an anechoic room), and then in the room to be examined. The measurement in the free field is done by measuring at a distance of 10 m from the sound source and is repeated for at least 5 measurements at various locations to create an average, [22, 9]. As in free field, the room to measure should have an equal amount of measurements made. The gain of a room can be described with the following equations from DS 3382-1, [9]:

$$G = 10 * log10 \left(\frac{\int_0^\infty p^2(t)\, dt}{\int_0^\infty p_{10}^2(t)\, dt} \right) = L_{pE} - L_{pE,10}[dB] \tag{2.62}$$

$$L_{pE} = 10 * log10 \left[\frac{1}{T_0} * \int_0^\infty \frac{p^2(t)\, dt}{p_0^2} \right] [dB] \tag{2.63}$$

$$L_{pE,10} = 10 * log10 \left[\frac{1}{T_0} * \int_0^\infty \frac{p_{10}^2(t)\, dt}{p_0^2} \right] [dB] \tag{2.64}$$

Where: $p(t)$ is the instantaneous sound pressure of the impulse response measured at a point in the room measured in Pascal, where 1 $Pascal = 1\frac{kg}{m*s^2}$, $p_{10}(t)$ is the instantaneous sound pressure of the impulse response measured at 10m in free field, p_0 is $20\mu Pa$, T_0 is 1 second, L_{pE} is the sound pressure exposure level of $p(t)$, and $L_{pE,10}$ is the sound pressure exposure level of $p_{10}(t)$.

If any of the rooms are not large enough to complete the measurements at 10 meters, they can instead be done at 3 meters and modified with:

$$L_{pE,10} = L_{pE,d} + 20 * log10(d/10)[dB] \tag{2.65}$$

Or even simpler by using, [22, 9]:

$$L_p = L_{p,10}[dB] \tag{2.66}$$

Where: d is distance in meters, L_p is the sound pressure level averaged across every measurement point, and $L_{p,10}$ is the sound pressure level measured at 10 m in the free field.

Alternatively, if a sound source with a known sound power level is available, the gain can be obtained by the following, [22, 9]:

$$G = L_p - L_W + 31[dB] \tag{2.67}$$

Where: L_p is the sound pressure level averaged across every measurement point, and L_W is the sound power level of the sound source, and should be measured according to DS 3741.

2.7.2 Reverberation Time 60 - RT60:

Reverberation **T**ime 60 describes the time it takes for sound a level to decay 60 dB. In many practical applications, background noise makes it difficult to measure a full 60 dB decay, and therefore, T20 or T30 is used instead. T20 and T30 differ from RT60 by measuring either 20 or 30 dB decay and following the description set in DS ISO 3382-2, the decay time must be measured from -5 to -25/30 dB, and not from 0 to -20/30 dB, [36, 22, 37, 38]. In figure 2.9a, an example of measuring T30 can be seen, where it should be noted that the decay is first measured from -5 dB. When using T20 or T30, the time measured should be multiplied by 3 or 2 to harmonize with RT60. It should furthermore be noted that T20 is most frequently used, as DS 3382-2 states that *"the subjective evaluation of reverberation is related to the early part of the decay"*, and that *"the signal-to-noise ratio is often a problem in field measurements, and it is often difficult or impossible to get an evaluation range of more than 20 dB."*, [36].

Sabine's equation for a diffuse sound field can be used to estimate a room's reverberation time. It takes the input V for the volume of the room in cubic meters, α for the absorption coefficient in $\left[\frac{sabin}{m^2}\right]$, and S for the surface area of the room in square meters, and is used as:

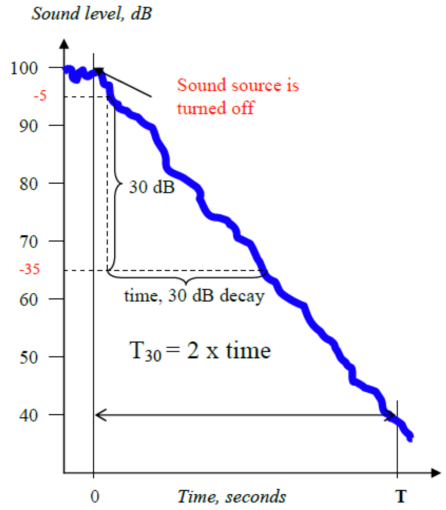
$$RT60 = \frac{0.161 * V}{S_i * \alpha_i} \leftrightarrow RT60 = \frac{0.161 * V}{A}\,[s] \tag{2.68}$$

Where:

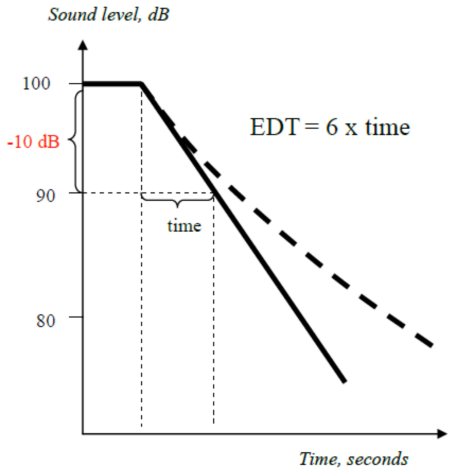
$$A = \sum_{i=o}^{i=\infty} S_i * \alpha_i \tag{2.69}$$

The reasoning behind equation 2.69/the lower fraction of 2.68 is to compensate for different materials with different absorption coefficients in a room. It should, however, be noted that since Sabine's formula assumes a diffuse sound field, it is best used concerning EDT rather than T30 or T20 due to the corresponding decay curves. This can be seen in figure 2.9b, where the solid straight line matches a continued approximation derived from the EDT. The dashed line is a more realistic decay curve of a room with absorbent materials, such as acoustic ceilings, [22]. In general, some optimal RT ranges are, [10, 11, 12, 22, 23]:

- Speech-oriented spaces, such as classrooms, meeting rooms, lecture halls, etc., require a short reverberation time of 0.3-1 second, to ensure speech intelligibility.
- Musical performance rooms, such as opera houses, concert halls, clubs, etc., would typically benefit from longer reverberation times, ranging from 1.5-3 seconds.
- Multi-use rooms such as dining halls, auditoriums, etc. benefit from a compromise, with reverberation times of 1.2-2 seconds.
- Sacred rooms/rooms of worship, such as churches, chapels, monasteries, etc., have much wider ranges, as they (dependent on religious purposes) have a reverberation time between 1-10 seconds, with gothic churches aiming at 6-13 seconds, [10].



(a) Definition of 30 dB decay (T30). Notice the timer starts at -5dB, [22].



(b) Illustration of EDT, [22].

Figure 2.9: T30 and EDT illustrations, [22].

2.7.3 Early Decay Time - EDT/T10:

Early decay time describes the first 10 dB decay within a room and is sometimes denoted T10, similarly to T20 and T30. The main difference from T20 and T30 is that EDT does not include the same -5 dB buffer but instead measures the actual first 10 dB decay. According to [22], [39], and DS 3382-1, [9], the EDT should be the subjectively most important reverberation characteristic when determining the acoustic properties of a room. EDT is determined by the first 10 dB of attenuation multiplied by 6 to reach a number comparable with RT60. The EDT equation is most beneficial when it comes to the psychoacoustic perception of a room and for determining "ideal" decay time, [39]. This is caused by the minimum of summed reflections present within the first 10 dB decay.

2.7.4 Clarity - C50/C80:

To determine the clarity relation of a room as an acoustical parameter, C50 or C80 is used. C50 estimates the reflections and their energy present after 50 ms, while C80 estimates them after 80 ms. The idea is that reflections present after the critical time limit muddy perceived speech and reduce clarity. The critical time limit varies from person to person, with age, and is furthermore dependent on what exactly is being listened to, [22, 10, 11, 23, 9]. C50 is most often used for speech intelligibility and C80 is used for music clarity. The C50 and C80 relation is calculated by:

$$C50 = 10 * log10 \left(\frac{Energy(0 - 50ms)}{Energy(51ms - end)} \right) [dB] \tag{2.70}$$

$$C80 = 10 * log10 \left(\frac{Energy(0 - 80ms)}{Energy(81ms - end)} \right) [dB] \tag{2.71}$$

Where the "Energy" values are derived from a broadband impulse response in the room. The impulse response is recorded and divided into integrated squared energy values before and after the critical time limit, [22, 10, 11, 23, 9]. Both can also be found generally by:

$$C_{t_e} = 10 * log10 \left(\frac{\int_0^{t_e} p^2(t)\, dt}{\int_{t_e}^{\infty} p^2(t)\, dt} \right) [dB] \tag{2.72}$$

Where: C_{t_e} is the early-to-late index. t_e is the early time limit (e.g., 50 or 80 ms) and $p(t)$ is the measured sound pressure of the impulse response, [9].

For speech clarity, values above 0 dB are considered good, while for music clarity, it depends on music type and preference, but in most cases, a value between -2 dB and +4 dB is considered acceptable, [23].

2.7.5 Definition - D50:

Definition/Deutlichkeit after 50 ms is a percentage describing how many percent of the total energy is present within the first 50 ms. D50 is strongly correlated to C50, as D50 (and vice-versa C50) can be found by, [9]:

$$C50 = 10 * log10 \left(\frac{D50}{1 - D50} \right) [dB] \tag{2.73}$$

Or by itself using:

$$D50 = \frac{\int_0^{0.050} p^2(t)\, dt}{\int_0^{\infty} p^2(t)\, dt} [\cdot] \tag{2.74}$$

2.8 Measuring Techniques

As mentioned many times throughout this report already, the best way to determine the acoustical parameters of materials or rooms is by measuring them. The method for measuring acoustical properties has been developed considerably over the last two decades, with the introduction of increasingly powerful computers, algorithms, and electroacoustical instruments.

2.8.1 Impulse Response of a Room

The impulse response of a room reveals everything there is to know about a signal's propagation from one point to another in a room if it is assumed to be time-invariant. The transfer function created from the Fourier transform of an impulse response yields the linear transmission system. Spatial and directional information requires a binaural impulse response. A regular impulse response, on the other hand, uses only one omnidirectional microphone and speaker. In contrast, a binaural impulse response requires a dummy head or a person with microphones inserted into each ear canal.

To be considered, the impulse response must be measured using distortionless components, and the room should be in the same state as when in use. The impulse itself is a short burst, often in the low millisecond range, excited with intensity high enough for decaying sound fields to be measured. The impulse can be created by various methods, such as pistol shots, wooden clappers, or an omnidirectional loudspeaker playing a known signal. By using the latter method, exact impulses can be designed and used. Another benefit of using the loudspeaker with a known impulse response is that the convolved signal can be broken into fragments, containing the impulse response of the speaker and the actual room impulse response. Mathematically, it can be written in the time domain as:

$$g'(t) = g(t) * g_L(t) \tag{2.75}$$

Where $g(t)$ is the room's impulse response, and $g_L(t)$ is the response of the loudspeaker. The deconvolution is best done in the frequency domain by Fourier transforming the response, yielding:

$$G'(f) = G(f) * G_L(f) \tag{2.76}$$

The signal exciting the speaker is given as $s't$ in the time domain and as $S'(f)$ in the frequency domain:

$$s'(t) = \int_{-\infty}^{\infty} s(\tau) * g(t-\tau)d\tau \tag{2.77}$$

$$S'(f) = S(f) * G(f) \tag{2.78}$$

From the above equations, the impulse response of the room ($g(t)$) can be found from the inverse Fourier of either equation 2.76 or 2.78. This method is known as dual channel analysis, as both the original signal and the speaker's impulse response is known, isolating $g(t)$ as the only unknown.

To remove uncertainties, measurements should be done several times to achieve mean values. With each increase in the number of measurements N, the energy of the impulse responses in total grows by N^2, while background noise and measuring noise only grow by N. By doing so, the SNR is increased by $10 * log_{10}(N)$.

2.8.2 Acoustical Properties of Materials

While the impulse response of a room is the most accurate way of determining the characteristics of a room, it relies on the room being built beforehand. To mitigate unwanted acoustical properties in the sketching phase, the room can be simulated, using software such as Odeon, COMSOL Multiphysics, etc., if the materials and schematics of the room have been finalized. Therefore, it is highly beneficial to know the acoustical properties of materials used in the room beforehand, such as sound absorption, reflection, transmission, and diffusion. Said properties can be determined by measuring impulse responses of the materials when placed in known environments, such as reverberation rooms or impedance tubes.

The room can be designed to meet acoustic benchmarks by acquiring these properties for all materials used. Unfortunately, measuring such materials is prone to faulty or unreplicable results, as quite a few aspects can alter them. For reverberation rooms, the main fault for material testing is the lack of strictly standardized sizes and forms. Some standards do exist though, as specified in DS/ISO 354:2003, such as a minimum volume V of $150m^3$, a recommended volume of $200m^3$, and a maximum volume of $500m^3$ to reduce the possibility of attenuation by air and a maximum length in a straight line anywhere in the room of $l_{max} = 1.9 * V^{\frac{1}{3}}$, [40]. Furthermore, diffusers should be placed in the room until the absorption coefficient reaches a maximum and remains there no matter how many diffusers are added - a rule of thumb is that the area equals 15-25% of the rooms total surface area, [40]. More rules exist for the reverberation room, but as can be seen by these general rules, a lot of variation is permissible, yielding in non-equal measurements of the same material, placed in different reverberation rooms.

Regarding the impedance tube, two methods currently exist: 1. the standing wave method. 2. The multimicrophone method. The standing wave method has been the industry standard for many years (since 1902, [21]), but has recently been replaced by the multi-microphone method (2023, [14]). Unfortunately, both methods have their flaws: Common for both methods is mounting of test specimens, as specimens are easily

subjected to deformation and/or being cut slightly in the wrong size due to the circular form, resulting in air gaps. Another issue present mostly for the standing wave method is the human error present when reading measurements off of analog instruments for each individual frequency. The multimicrophone method is as prone to user error as it is essential that microphones are calibrated correctly before each test.

Depending on the purpose of the material test, the impedancetubes are limited to the frequencies for which they can test. The Brüel & Kjær 4002 standing wave apparatus has working frequencies ≈90-6500Hz, and the type 4206 multimicrophone impedance tube has ≈50-6400Hz. However, these working frequencies are dependent on the attached tube, resulting in two different tubes for use in different frequency bands. While both impedance tube types cover the frequency band most influential on human hearing, they still leave a rather large frequency band untested.

2.8.3 Problematic Frequencies

As can be extrapolated from the report so far, many acoustical models exist for behavior from 100Hz to 6kHz, but outside of this band, models and results become increasingly experimental. This leaves room for improvement on the remaining spectrums, especially as background noise often has components in these regions. Low-frequency noise has been an increasing health focus worldwide and would benefit from being attenuated, both at the source and within rooms. While most low-frequency noise is vibrational, large amounts are still audible. The infrasound spectrum (0-20Hz) is especially interesting, as most levels are not audible, but felt instead. Despite being largely inaudible, infrasound can still have psychological and physiological effects on individuals, potentially causing discomfort or even health problems, especially at higher amplitudes. This makes the study and mitigation of infrasound an important area of research. Therefore, extending research and testing to cover these problematic frequencies, particularly in the infrasound range, would help create more comprehensive acoustical models that account for the full spectrum of human hearing and environmental noise. Addressing this gap could lead to better control of noise pollution, especially in environments where noise exposure is a concern for health and well-being.

2.9 Problem Statement

Based on the problem analysis/dive into room acoustics, several statements can be made when reviewing the initial problem statement:

"How can "good acoustics" be defined and quantified in a way that balances theoretical ideals with real-world applicability?"

The analysis shed light on the main subject of quantizing "good" acoustics, before embarking in an effort to describe acoustical phenomena by math and intuitive relations. The immediate characteristics for obtaining good acoustics were D50, C50, and most importantly RT60 and EDT. However, these properties were inherent by a room, and could be hard to quantify, if material properties were unknown. The most influential properties were found to be: Reflection, diffusion, and absorption coefficients, wall impedance/admittance, transmission loss and resonance frequency, as shown in table 2.2.

With common acoustical measures and wall properties in place, room acoustics in relation to modal planes and image sources was investigated, leading to the possibility of describing soundfields using ray-tracing, rather than spherical waves. This idealization enables easier computation of modeled reverberation times, while also making it possible to predict orientation of reverberated sound rays.

However, a prevailing issue began appearing while analyzing room acoustics as a whole. When it came to absorbers and impulse responses of rooms, it became clear that many models rely on data for 100-6000Hz rather than the full audible spectrum, or at least the spectrum used for equal loudness curves (20-12500Hz). With increasing amounts of research pointing towards psychological and physiological concerns regarding especially low frequency sounds. It would pose an interesting task to determine acoustical properties below 100Hz, and preferably entering the infrasound band for optimal modeling.

The low-frequency predicament is found in impedance tubes and reverberation rooms as well. This adds to the inherent problems of absorption coefficient measurements of materials in impedance tubes and especially reverberation rooms mentioned in 2.8.2. Due to the unreplicable nature related to reverberation rooms, impedance tubes would be a better starting point for developing a system capable of determining acoustical properties. The main issue of impedance tubes is the circular form, as samples are hard to create without risking deformation and/or air gaps. Therefore, this project's problem statement is:

"How can a square impedancetube capable of measuring very low frequency properties of construction materials be developed?"

3 Technical Analysis

In an effort to analyze the technical needs of the project, common methods for measuring acoustic properties are examined in depth. Furthermore, signal processing procedures for easing acoustical measurements will be examined as well, to gain insight into designing a complete system capable of measuring wideband frequency properties, along with the focus on low frequencies given by the problem statement.

When measuring the acoustic properties, the R_{10} series of the international "preferred numbers" should be used, based on the standardized Renard numbers, [32]. The frequencies used closely match the center frequencies of $\frac{1}{3}$ octave bands. Octave bands are described in appendix E on page 122, and the complete table of preferred Renard numbers applied in acoustics can be found in appendix F on page 125.

3.1 Reverberation Room Method

The reverberation room is a common method used to test large samples, such as entire rows of seats from a concert hall, resonating panels, entire rockwool batts, furniture, large-scale absorbers, edge absorption, etc. The reverberation room functions by following the dimensions specified in DS/ISO 354:2003, and placing "hard" walls with little to no absorption at all edges, [40]. To prolong the reverberation time, diffusers are placed within the room, minimizing the amount of parallel surfaces and scattering sound energy in all directions.

However, as pointed out in several publications by different authors, the reverberation room is not a trustworthy source of absorption/reflection coefficients, in spite of semi-standardized room configurations, [32, 34, 33, 35, 31, 7, 8, 41, 42, 17, 5].

3.1.1 Measurement of Sound in a Reverberation Room - DS 354:2003

As briefly mentioned in "Acoustical Properties of Materials", reverberation rooms are standardized through DS/ISO 354:2003. The room itself should as a minimum have a volume V of $150m^3$, a recommended volume of $200m^3$, and a maximum volume of $500m^3$ to reduce the possibility of attenuation by air and a maximum length in a straight line anywhere in the room of $l_{max} = 1.9 * V^{\frac{1}{3}}$, [40]. The number of diffusers present within the reverberation chamber is given by Annex A in DS/ISO 354:2003. The diffusers should be sheets of different sizes between 0.8-$3m^2$ (single side) with a mass of $5\frac{kg}{m^2}$, [40]. When finding the threshold of diffusivity, a sample of 5-10cm thick homogeneous, porous material with an absorption coefficient over 0.9 in the frequency range of 0.5-4kHz is placed in the room. The room is then tested without diffusers, and thereafter with increases of $5m^2$ until the absorption coefficient reaches a constant value in spite of an increasing number of diffusers mounted. For rectangular rooms, a diffuser area (both sides) matching 15-25% of the total surface area of the room is sufficient, [40].

When testing plane absorbers in a reverberation room, they should be 10-12m^2 for rooms with a volume of 200m^3. For larger rooms, the test specimen area should increase with a factor of $(\frac{V}{200})^{\frac{2}{3}}$. Furthermore, the plane absorbers should be mounted according to Annex B in DS/ISO 354:2003. Discrete objects such as furniture, free-standing objects, or similar, should be installed in the reverberation rooms in a similar fashion, as when installed in practice. When testing discrete objects, several samples should be placed at a distance of 2m between them. For both plane absorbers and discrete objects, the distance to any boundaries in the chamber should be 1 meter or more, unless otherwise specified by the installation manual.

To ensure the reverberation room complies with the standard, table 3.1 is used. The equivalent sound absorption area is a descriptor of "*hypothetical area of a totally absorbing surface without diffraction effects which, if it were the only absorbing element in the room, would give the same reverberation time as the room under consideration*", and is given by equation 3.1 where V is the volume of the empty reverberation room, c is the propagation speed of sound in air, T_1 is the reverberation time of the empty reverberation room, and m_1 is the power attenuation coefficient calculated according to ISO 9613-1, [40, 43]. The values given for equivalent sound absorption area are the maximum permissible value for the reverberation room to be accepted as per the standard.

$$A_1 = \frac{55.3 * V}{c * T_1} - 4 * V * m_1 \tag{3.1}$$

Frequency, Hz	100	125	160	200	250	315	400	500	630
Equivalent sound absorption area, m^2	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
Frequency, Hz	800	1,000	1,250	1,600	2,000	2,500	3,150	4,000	5,000
Equivalent sound absorption area, m^2	6.5	7.0	7.5	8.0	9.5	10.5	12.0	13.0	14.0

Table 3.1: Maximum equivalent sound absorption areas for room volume $V = 200m^3$ for $\frac{1}{3}$ octave bands with given centerfrequencies, [40].

3.1.2 Subtleties of Using a Reverberation Room

While widely accepted as the standard measurement method for acoustic properties of large specimens, the reverberation room method poses several subtleties that should be considered. The main consideration should be that no reverberation rooms are identical, leaving measurements as a value only obtained in that specific room. Effectively, this renders measurements more or less useless, as they are not replicable in different rooms, and possibly not even in the same room.

Moreover, if samples are mounted slightly differently in testing, they might not produce a similar result when installed at the job site. Orientation of samples, along with distances between samples, will also yield different results. A factor largely contributing to this is the increased absorption at the edges of porous materials. In general, the reverberation room is great for achieving a vague idea of a sample's acoustic properties, but values are rarely applicable outside of that exact test environment, [32, 34, 33, 35, 31, 7, 8, 41, 42, 17, 5].

3.2 Standing Wave Method

The standing wave method has been the go-to method of obtaining consistent and accurate acoustical properties for decades. It is considered the most reliant method for assessing reflection and absorption coefficients of materials. However, the standing wave method has significant drawbacks in the time needed to perform measures for large frequency bands, while also being incapable of achieving correct measurements of resonance absorbers in most cases.

To comprehend the standing wave method, the DS 10534-1 2001, Brüel & Kjær Application Manuals, and transmission line theory will be utilized, [13, 32, 33, 34, 35]. But first, a graphical example/analogy will be introduced:

3.2.1 Graphical Analogy

Impedance tubes utilizing the standing wave method are often portrayed as seen in figure 3.1 or similarly, most resembling SWR patterns as seen in transmission line theory. However, it can be quite difficult to get a constructive idea of what is happening physically from such an image. Therefore, figure 3.3 is used instead. This figure describes the incident sound wave by equation 3.2, the reflected sound wave by equation 3.3, and the summed waveform by equation 3.4. Now, the way that a drawing such as 3.1 should be interpreted is, that each "hoop" is the upper part of both waveforms. This can be seen in figure 3.2 where the upper half of the waves can be seen, with x_{min1}, x_{min2}, and x_{max} marked as well. This interpretation can be made, as the reflected wave is phase shifted 180°, meaning that its upper maximum is equivalent in position to the lower maximum of the incident wave. This also means that the intersection between the waves matches minimums.

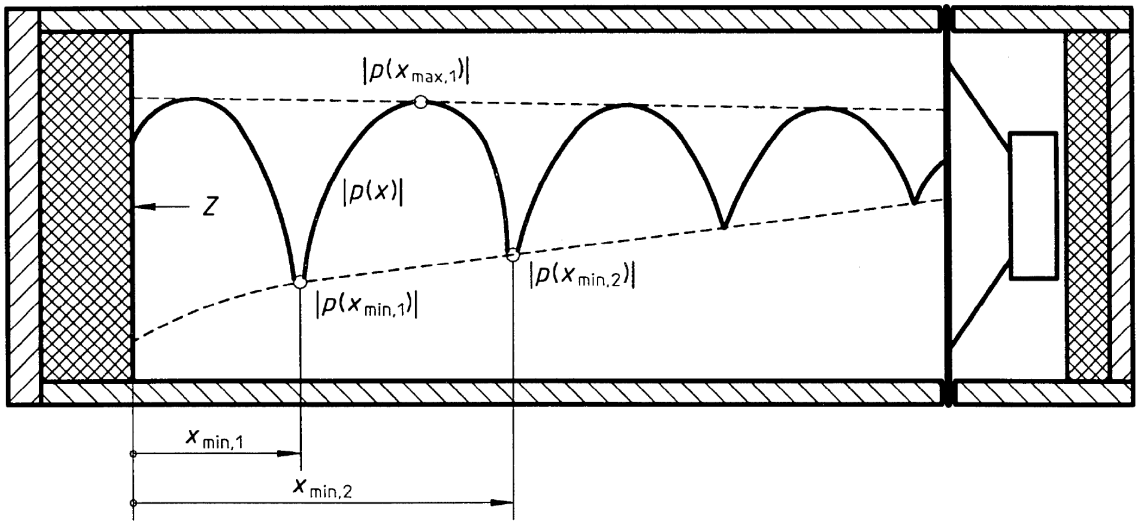


Figure 3.1: Typical graphic representation of the standing wave pattern in an impedance tube, [13].

$$p_i(x) = A * cos(kx - \omega T) \tag{3.2}$$

$$p_r(x) = r * A * cos(-kx - \omega T + \pi) \tag{3.3}$$

$$p_t(x) = A * cos(kx - \omega T) + r * A * cos(-kx - \omega T + \pi) \tag{3.4}$$

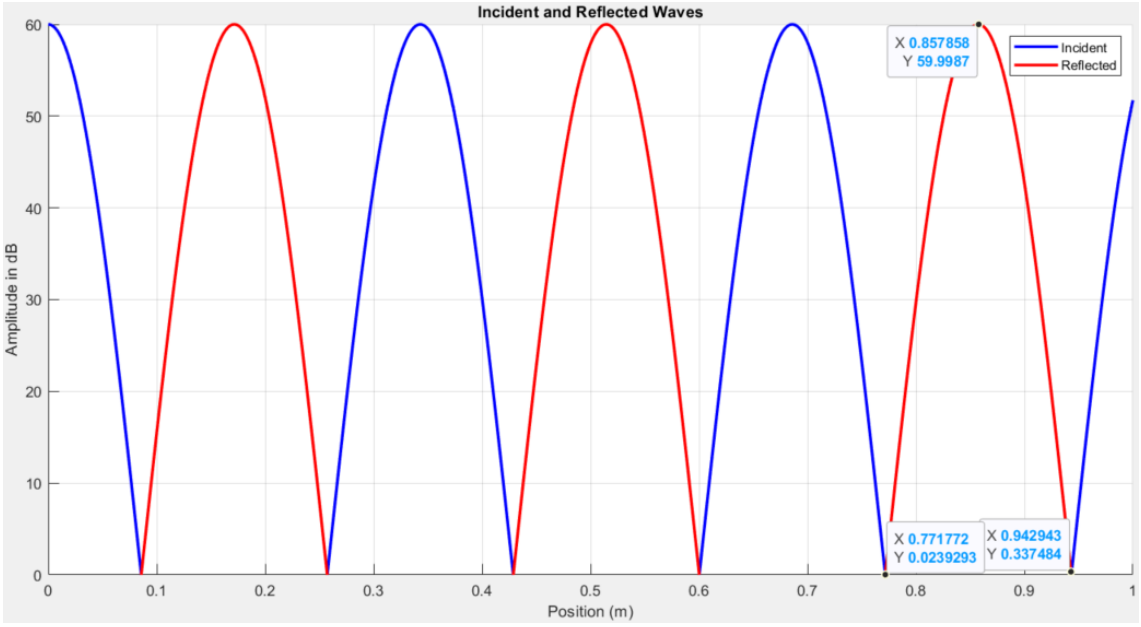
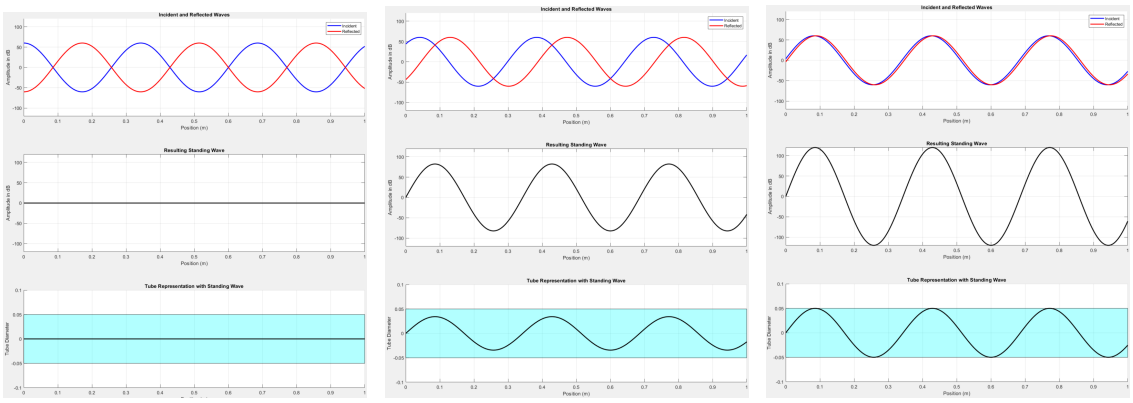


Figure 3.2: Interpretation of standing waves in an impedance tube, with x_{min1}, x_{min2}, and x_{max} marked.



(a) (b) (c)

Figure 3.3: Standing waves visualized as incident, reflected, and summed waveform as well as the summed waveform in an impedance tube. The blue waveform is the incident wave, and red is the reflected wave. All figures have the tube-end to the right on the x-axis in meters, and the two top illustrations have amplitude in dB on the y-axis, with the lower blue illustration having tube diameter in meters on the y-axis.

3.2.2 Standing Wave Method Standard - DS 10534-1-2001

To obtain qualitative measurements using an impedance tube, DS 10534-1-2001 should be used as a guideline. The standard describes all related variables and how they should be used, along with defining the physical aspects that an impedance tube should meet. While the standard introduces significant formulas, the related math will be explained in section 3.2.3 starting page 32. Therefore, this section will mostly relate to the physical characteristics of an impedance tube, such as diameter, length, material, etc. This also means that the main specifications for the use of an impedance tube are best found in the standard itself, and will be left out of this section.

As the name suggests, an impedance tube should be tubular, but is not restricted in shape, and could therefore be circular or rectangular. The first parameter to acquire is the length of a tube, given by equation 3.5, with λ being the desired frequency wave-

length to test. It should however be noted that this equation yields a length far greater than the one used in the Brüel & Kjær Type 4002, which will be explained in greater detail in section 3.2.3 on page 32. When defining the diameter, equation 3.6 is used to find the frequency diameter boundaries when length is known, and equation 3.7 is used to find the upper boundary for rectangular tubes, and 3.9 for circular tubes. In equations 3.6, 3.7, and 3.9 the variables f and λ are interchanged with a given value for a specific frequency. Equations 3.8 and 3.10 are used to establish the upper frequency threshold.

The equations can only be used by specifying variables, such as the desired frequency boundaries or physical sizes, to determine the corresponding frequency boundaries. By investigating the equations, it also becomes clear that for rectangular tubes, a square form is desired in most cases, as it is the longest side which determines the upper frequency boundary.

$$l \geq \frac{3 * \lambda}{4} \tag{3.5}$$

$$l = \frac{250}{f} + 3 * d \rightarrow d = \frac{l}{3} - \frac{250}{f} \tag{3.6}$$

$$d \leq 0.5 * \lambda \tag{3.7}$$

$$f_U * d \leq 170 \tag{3.8}$$

$$d \leq 0.58 * \lambda \tag{3.9}$$

$$f_U * d \leq 200 \tag{3.10}$$

The tube itself should be made of a rigid and smooth material that will not be excited by vibrations. Furthermore, the tube must be straight within 0.2%. The material which the tube is made from is effectively not as important as the smoothness and rigidity, however, metal or concrete tubes will be easier to create uniform surfaces with. Rectangular tubes will especially have extra emphasis on the rigidity of the structure and smoothness in corners, as the form itself is not as rigid as circular tubes.

The sample holder can be constructed as either an integrated terminating end with room for a sample in front, or by making the end removable. With the terminating end, a section of the tube must be removable, to insert a sample. The removable section should be long enough to enable the possibility of testing cavities behind a material sample. By using a removable end, the end section must still conform to the other specifications of the tube, while also enabling the possibility of inserting a sample into the end. When mounting samples, seals should be made airtight by elastic gaskets, such as vaseline.

The electrical equipment necessary to facilitate use of an impedance tube is:

- A moveable microphone
- Signal generator
- Signal processing equipment
- Loudspeaker
- Thermometer

It should go without saying that each of these tools should be of sufficient quality to make measurements viable. Specific relations between the equipment are best found by reading the standard, [13].

3.2.3 Math Behind the Standing Wave Method

This section outlines key equations used for the standing wave method. Depending on which source material is used, some formulas differ but essentially calculate the same, exemplified by the equations for SWR in 3.11 and absorption coefficient in 3.12, [13, 32, 33, 34, 35].

$$n = \frac{(A+B)}{(A-B)} = \frac{p_{max}}{p_{min}} = \frac{1+|r|}{1-|r|} = \frac{N}{\rho_0 * c} \tag{3.11}$$

$$\alpha = 1 - \frac{B^2}{A^2} = 1 - \left(\frac{n-1}{n+1}\right)^2 = \frac{4}{n+\frac{1}{n}+2} = 1-|r|^2 = \frac{4}{\frac{(n+1)}{(n+2)}} = \frac{4n}{n^2+2n+1} \tag{3.12}$$

Where:

- n is the standing wave ratio
- A is the measured amplitude of the incident sound wave
- B is the measured amplitude of the reflected sound wave
- p_{max} is the maximum sound pressure given by equation 3.19
- p_{min} is the minimum sound pressure given by equation 3.20
- r is the reflection coefficient of a sample given by equation 3.14
- N is the "true impedance" given by equation 3.13
- ρ_0 is the static gas density value ($\approx 1.2[\frac{kg}{m^3}]$ for air)
- c is the speed of sound (set to $343[\frac{m}{s}]$

The SWR n is used to determine how much of the wave is reflected from a material by measuring the difference between minima and maxima of the standing wave, which can be found by equations 3.20 and 3.19. It is not a percentage but a dimensionless variable used to determine the reflection coefficient r given in equation 2.15 and/or the absorption coefficient α.

$$n = \frac{N}{\rho_0 * c} \leftrightarrow N = n * \rho_0 * c \tag{3.13}$$

$$|r| = \frac{n-1}{n+1} \tag{3.14}$$

With the reflection coefficient known, it is possible to calculate the reflected wave as shown in equation 3.3. Another possibility is measuring the reflected wave at a given point as in 3.16. The incident sound wave is given by equation 3.15.

$$p_i = A * cos(2 * \pi * f * t) \tag{3.15}$$

$$p_r = B * cos(2 * \pi * f * \left(t - \frac{2y}{c}\right) \tag{3.16}$$

where p_i is sound pressure of the incident sound wave in Pa, p_r is sound pressure of the reflected sound wave in Pa, f is frequency of excitation in Hz, y is distance of observed point from the surface of the sample in m, c is velocity of sound within the tube in m.s-1, t is time in seconds.

By adjoining these equations, the total pressure at point y can be found by equation 3.17. Unto which the addition theorem shown in equation 3.18 can be used to yield the minimum and maximum pressures in equations 3.20 and 3.19, which then ultimately can be used to determine the reflection/absorption of a material.

$$p_y = p_i + p_r = A * cos(2 * \pi * f * t) + B * cos(2 * \pi * f * \left(t - \frac{2y}{c}\right) \tag{3.17}$$

$$cos(\theta - \phi) = cos(\theta) * cos(\phi) + sin(\theta) * \sin(\phi) \tag{3.18}$$

$$p_{y_{max}} = (A + B) * cos(2 * \pi * f * t) \tag{3.19}$$

$$p_{y_{min}} = (A - B) * cos(2 * \pi * f * t) \tag{3.20}$$

In section 3.2.2 it was mentioned that the dimensions of the tube could be determined differently, explaining how the standing wave apparatus type 4002 can rely on a tubelength of only a meter, instead of 2.8583 meters, as is calculated by using equation 3.5. In the instruction manual for the 4002 impedance tube revised in 1979, [34], it is stated that length of the tube must be at least $\frac{\lambda}{4}$ for low frequencies and the diameter $0.586 * \lambda$ for higher frequencies, virtually identical to 3.9. By using $\frac{\lambda}{4}$ to determine the lower boundary, it is seen that the type 4002 should be capable of measuring frequencies down to 85.75 Hz with a length of 1 meter. This matches the specification plus some overhead for the type 4002, as it specified to measure down to 90 Hz. When the frequency no longer shows isolated maximums, the pressure in front of the sample should be used.

3.2.4 Transmission Line Theory - Smith Chart

In transmission line theory, impedance, admittance, and Smith charts are fundamental for analysis of wave propagation in networks, especially at increasingly high frequencies. The standing wave method in acoustics has many similarities to the standing wave ratio in transmission line theory. In transmission line theory, SWR is used to impedance-match antennas and transmission lines, minimizing lost power. In acoustics, something similar is happening, as acoustical impedance can reveal where in a material reflections are happening, thus making it possible to tune materials to absorb/reflect exact frequencies. The normal acoustic impedance of a material is given as Z_n and describes the sound pressure at the surface and its associated particle velocity, [34].

$$Z_n = \frac{p}{v} \tag{3.21}$$

Which for the standing wave tube becomes:

$$Z_n = \frac{p_i + p_r}{v_i + v_r} \tag{3.22}$$

As the characteristic impedance of air ρ_0 influences both p_i and p_r, the relations become:

$$p_i = \rho_0 * c * v_i \; and \; p_r = \rho_0 * c * (-v_r) \tag{3.23}$$

$$Z_n = \left(\frac{p_i + p_r}{p_i - p_r}\right) * \rho_0 * c \tag{3.24}$$

The equation for p_r can be rewritten into:

$$p_r = |r| * p_i * e^{j*\Delta} \tag{3.25}$$

Where Δ is the phase angle between incident and reflected sound pressures. This rewrite can be inserted into equation 3.24 yielding equation 3.26 and further derived equation 3.27:

$$Z_n = \left(\frac{1 + |r| * p_i * e^{j*\Delta}}{1 - |r| * p_i * e^{j*\Delta}}\right) * \rho_0 * c \tag{3.26}$$

$$Z_n = (Re(Z_n) + jIm(Z_n)) * \rho_0 * c \tag{3.27}$$

The real and imaginary parts of 3.27 are given by, [34]:

$$Re(Z_n) = \frac{1 - r^2}{1 + r^2 - 2 * r * cos(\Delta)} \tag{3.28}$$

$$Im(Z_n) = \frac{2 * r * sin(\Delta)}{1 + r^2 - 2 * r * cos(\Delta)} \tag{3.29}$$

As the only variables are the reflection coefficient and the phase angle, normal acoustic impedance can be calculated by equations 3.28 and 3.29. The phase angle Δ is given by equation 3.30, where y_1 and y_2 are the distances from the sample material to minima 1 and 2, as can be seen in figure 3.4. It is important to remember the distance y_0 as well, as both equations in 3.30 otherwise will return 0.

$$\Delta = \left(\frac{4 * y_1}{\lambda} - 1\right) * \pi \leftrightarrow \Delta = \left(\frac{2 * y_1}{y_2 - y_1} - 1\right) * \pi \tag{3.30}$$

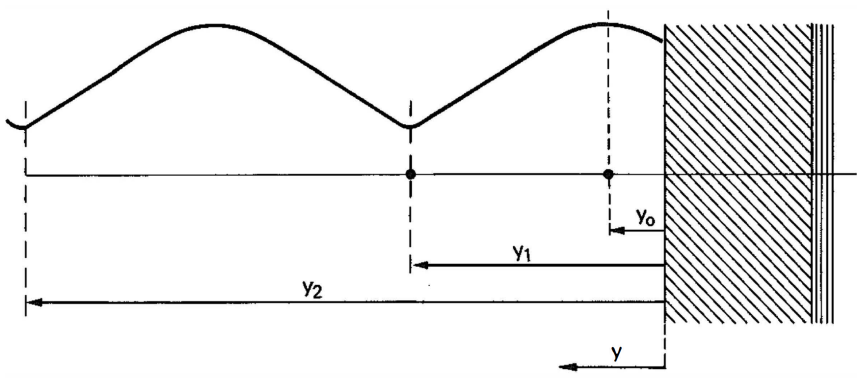


Figure 3.4: Diagram of where to find y_1 and y_2, [34].

When Δ has been calculated, its value must be converted from radians to degrees to make use of the Smith chart seen in figure 3.5. To use the Smith chart, a circle is first

drawn from the center with radius equal to the reflection coefficient value r. When that is done, the degree notation around the periphery of the Smith chart reveals which circle should be followed. The Smith chart in use can be seen in figure 3.6, where the green circle matches the value for a reflection coefficient of 0.3, and the phase difference is 120 degrees, shown by the red fractional circle. The blue and orange cross signifies where Re and Im components should be read off of. The values are read off as $0.88 + j0.58$ and $1.52 + j0.58$. Equations 3.31 and 3.32 are used to double check the interpretation, and determine which value is correct.

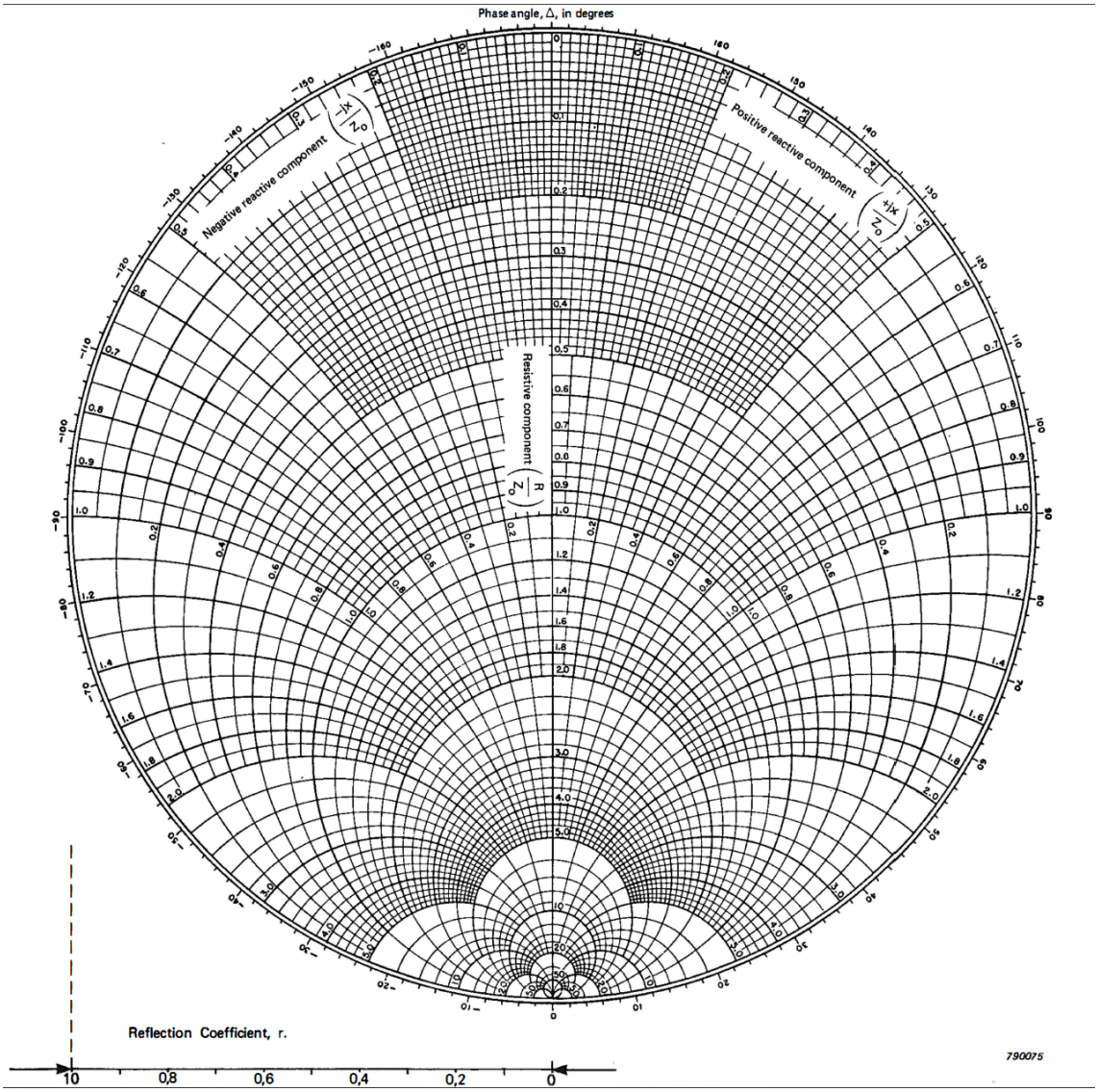


Figure 3.5: Acoustic smith chart, [34].

$$Re(Z_n) = \frac{1 - 0.3^2}{1 + 0.3^2 - 2 * 0.3 * cos(120)} = 1.5129 \tag{3.31}$$

$$Im(Z_n) = \frac{2 * 0.3 * sin(120)}{1 + 0.3^2 - 2 * 0.3 * cos(120)} = 0.57917 \tag{3.32}$$

As can be seen from the equations, the orange cross marks the correct spot within a small error margin. Graphically, the choice cannot be explained, as vanishingly little

information can be found regarding use of a Smith chart for acoustic impedance matching, with most coming from Per Brüels own literature, [32, 34, 33]. Nevertheless, Z_n can now be given as:

$$Z_n = (1.5129 + j0.57917) * \rho_0 * c \tag{3.33}$$

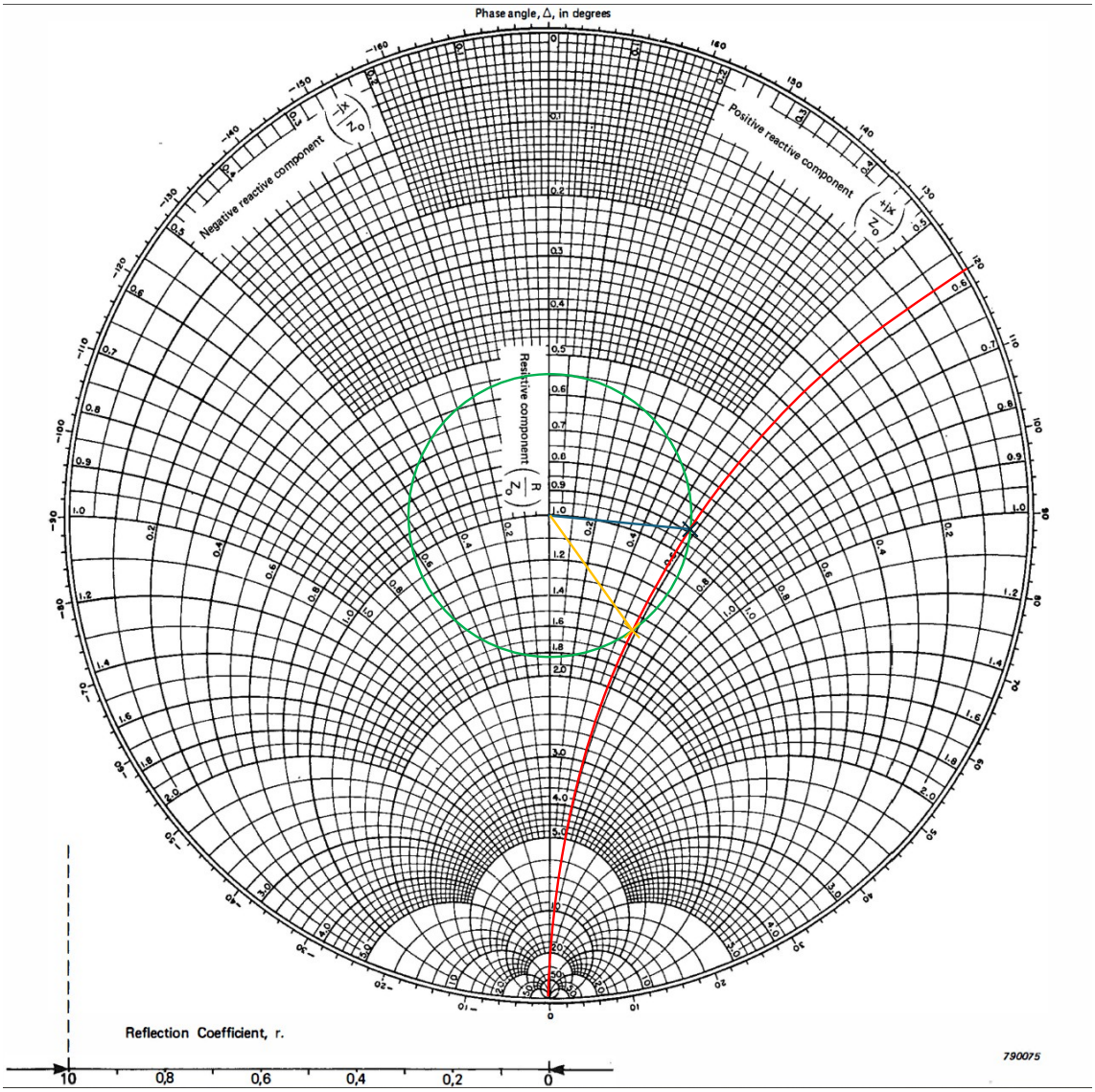


Figure 3.6: Acoustic smith chart in use.

By finding the impedance of a material, it is possible to find the reflection from equation 3.24 and the admittance of a wall by equation 2.21. More importantly, the impedance qualities can reveal where in a material reflection happens for specific frequencies and how sound travels within the material. Depending on the mismatch between air's characteristic impedance given by $\rho_0 * c$, the reflection will happen at different depths, [34, 44, 30]. The mismatch is given as a ratio of $\frac{Z_n}{\rho_0 * c}$, effectively revealing the admittance, as in equation 2.21, [7, 8, 34, 44, 30]. A larger impedance mismatch results in reflection happening closer to the surface, just as in transmission line theory. Therefore, if an impedance match can be made, the material will be completely admittant, which should not be confused with absorption. This can be used if specific frequencies should be attenuated, and others should be reflected as is. By inspecting the reactive j com-

ponent, it can be determined if the material acts as a mass or a spring, with negative reactance being spring-like behavior, and positive being mass-like behavior. A perforated panel absorber with an air gap behind it will typically have a negative reactance at certain frequencies, meaning that the sound pressure builds up inside the cavity. A thin metal sheet or a dense vibrating membrane can have a positive reactance at certain frequencies, meaning it behaves like a mass, slowing down and reflecting sound waves at those frequencies.

3.3 Multi Microphone Method

In short, the multimicrophone method (MMM) is very similar to the standing wave method, as it also employs a homogenous rigid tube, with a speaker at one end and a sample at the other. The main difference is the lack of probing microphones and the possible addition of a secondary tube for transmission measurements. The MMM has several benefits in comparison with the standing wave method. The first and foremost is the independence of having $\frac{1}{4}\lambda$ as the minimum length of a tube, which enables measurements at frequencies lower than possible in a standing wave impedance tube. The digital signal processing capabilities inherent to the MMM also enables the possibility of obtaining values for a frequency spectrum a lot faster than with the standing wave method.

The MMM utilizes the transfer matrix method, which is known from the optical domain as well. The transfer matrix method enables the determination of transfer functions of different materials when placed in a known environment. Impedances, absorption, and reflection coefficients can be found in these transfer functions.

3.3.1 Two Microphone Technique for Normal Sound Absorption Coefficient and Normal Surface Impedance Standard - DS 10534-2-2023

As for the standing wave method, international standards ensure that equal measurements can be made. For the MMM, the current related standards are DS 10534-2-2023 and ASTM E2611-24. Unfortunately, only the current DS standard has been available for this project, along with a dated version of the ASTM standard. However, from the dated ASTM standard, it can be derived that the requirements set by it are nearly identical to those set by DS, [14, 45, 46].

As in section "Standing Wave Method Standard - DS 10534-1-2001", the following section will be mostly related to the physical characteristics of an impedance tube utilizing the MMM. The mathematical background for deriving coefficients from the MMM is given in section 3.3.2.

Impedance tubes using the MMM can be made circular and rectangular, just as when using the SWT. Furthermore, several physical equations are similar to those of the SWT. When using the MMM, the tube length is dictated by the diameter, which in turn is still dictated by the upper frequency to measure. To determine the length of the impedance tube, it is necessary to know the distance between microphones, the distance from the speaker to the first microphone, and the distance from the second microphone to the sample, which is determined by the diameter. In equation 3.34, the diameter is given as a function of the shortest wavelength. Equation 3.35 reveals the maximum distance between microphones s in meters, whereas equation 3.36 yields the recommended minimum distance between microphones, given by 1.5% of the lowest frequency wavelength,

[14]. In equation 3.37, the complete range of distances is given as S, wherein all values can be chosen. It should however be noted, that larger distances between microphones increase accuracy for low frequencies, whereas small distances increase accuracy for high frequencies. Therefore, a higher bandwidth with high accuracy can be obtained by using multiple microphone positions.

$$rec = d \leq 0.5 * \lambda_U \ , \ circ = d \leq 0.58 * \lambda_U \ [m] \tag{3.34}$$

$$f_U * s \leq 0.45 * c \ [m] \tag{3.35}$$

$$0.015 * f_L \ [m] \tag{3.36}$$

$$S = 0.015 * f_L \leq f_U * s \leq 0.45 * c \ [m] \tag{3.37}$$

When determining the length of the impedance tube, the first microphone should be placed at least $\frac{1}{d}$ away, but ideally be $2 * d$ from the sample. Additionally, it is recommended that the microphone farthest from the sample be at most 250 mm away. From equation 3.38, the possible placements of the microphone nearest to the sample can be derived. The microphone closest to the speaker must be at least 3 diameters away from the speaker, which is calculated in equation 3.39. Assuming that the recommended distance of 250mm from the sample to the microphone farthest away is followed by placing the first microphone 250mm from the sample, the minimum length of an impedance tube is given by equation 3.40. However, the sample size still has to be added to the equation as well, yielding the actual minimum length to be given by equation 3.41. However, neither the DS nor the dated ASTM standard notes any maximum lengths of the impedance tube, meaning that the length can be adapted to fit the standing wave method as well, as a possible variation measured by multiple microphones instead of a movable microphone.

$$2 * d_{highest} \leq 0.25 - S_{highest} \tag{3.38}$$

$$min_D = 3 * d_{highest} \tag{3.39}$$

$$0.25 + min_D \tag{3.40}$$

$$0.25 + sample_{size} + min_D \tag{3.41}$$

Many of the same physical constraints for the material choice is similar as to the SWT, such as straightness within 0.2%, rigid construction, terminating end, etc. The main difference in setup is found in the electrical components, as the MMM utilizes microphones placed at sidewalls and some sort of DSP to process the measurements. Otherwise, equipment such as the signal generator, loudspeaker, and thermometer is still necessary.

When placing the microphones, they should be flush with the interior surface of the tube or slightly recessed. The placement of the microphone center must be known within 0.2mm. To ensure complete air-tightness, vaseline should be used in the threads for the microphone.

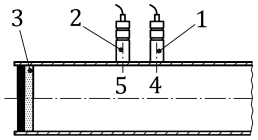
As with the SWT, the sample holder should be large enough to contain the desired samples, and otherwise follow the requirements of the tube itself. When inserting samples, sealants such as vaseline should be used on all joining surfaces.

The specific relations between equipment, algorithms, materials, etc. are best found by reading the standards, [14, 45].

3.3.2 Math Behind the Multi Microphone Method

Obtaining reflection/absorption coefficients using the MMM can be done in several ways, but all will rely on a measured impulseresponses. To derive correct transfer functions, two possibilities exist: interchanging microphones for repeated measurements or calculating a calibration factor. Both methods will be discussed.

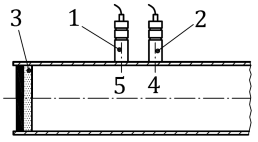
When interchanging the microphones, it is meant that the microphones physically is interchanged, as shown in figure 3.7. It is of utmost importance that the microphones are placed identically when interchanging, as differences in protrusion etc. can lead to faulty measurements.



Key

1 microphone A
2 microphone B
3 test specimen
4 position 1
5 position 2

(a) Standard configuration of microphone placements used to measure $H1_{12}$, [14].



Key

1 microphone A
2 microphone B
3 test specimen
4 position 1
5 position 2

(b) Interchanged configuration of microphone placements used to measure $H2_{12}$, [14].

Figure 3.7

In each configuration, a transfer function is measured, denoted $H1_{12}$ and $H2_{12}$. With transfer functions measured for both configurations, an average can be calculated as shown in equation 3.42. However, that version is only applicable if the signal processing equipment is capable of measuring transfer functions from mic A to B, and interchangeably from B to A. If that is not possible, equation 3.43 is used instead, as it assumes measurements can only be made from mic A to B. Both equations yield the transfer function H_{12}. The subscript notes that the transfer function is measured between two physical places, per the standard, [14].

$$H_{12} = (H1_{12} * H2_{12})^{\frac{1}{2}} = |H_{12}|e^{j\phi} \tag{3.42}$$

$$H_{12} = (H1_{12}/H2_{12})^{\frac{1}{2}} = |H_{12}|e^{j\phi} \tag{3.43}$$

If the user would like to reduce the amount of interchanging microphones, a calibration factor can be calculated instead. For this, an absorptive sample is inserted in the tube, and as before, transfer functions are measured for both configurations. Equation 3.44 is used when measurements can be made both ways, and 3.45 when only one-way measurements are possible.

$$H_C = (H1_{12}/H2_{12})^{\frac{1}{2}} = |H_C|e^{j\phi_c} \tag{3.44}$$

$$H_C = (H1_{12} * H2_{12})^{\frac{1}{2}} = |H_C| e^{j\phi_c} \tag{3.45}$$

With the calibration factor in place, all subsequent measurements can be corrected using equation 3.46. To use equation 3.46, subsequent measurements must be made with microphone configuration 1. This yields equation 3.47, where the hat notation marks uncorrected values for the transfer function, phase, real, and imaginary parts.

$$H_{12} = |H_{12}| * e^{j*\phi} = \frac{\hat{H_{12}}}{\hat{H_C}} \tag{3.46}$$

$$\hat{H_{12}} = |\hat{H_{12}}| * e^{j*\hat{\phi}} = \hat{H}_r + j * \hat{H}_i \tag{3.47}$$

The complex transfer function between the two microphone locations is given by the following three equations:

$$H_{12} = \frac{S_{12}}{S_{11}} = |H_{12}| e^{j\phi} = H_r + j * H_i \tag{3.48}$$

$$H_{12} = \frac{S_{22}}{S_{21}} = |H_{12}| e^{j\phi} = H_r + j * H_i \tag{3.49}$$

$$H_{12} = \left[\frac{S_{12}}{S_{11}} * \frac{S_{22}}{S_{21}}\right]^{\frac{1}{2}} = |H_{12}| e^{j\phi} = H_r + j * H_i \tag{3.50}$$

Where 3.48 is best used when there is noise at the output (signal 2), 3.49 is used when there is noise at the input (signal 1), and 3.50 is used when noise is present at both the input and output. S_{Xx} denotes the complex sound pressure P_X at microphone position x.

If deterministic measurements signal such as chirps or MLS, are used, the transfer function can be estimated using equations 3.51 and 3.52. Equation 3.51 is the frequency domain deconvolution based on FFTs, and equation 3.52 is based on impulse responses.

$$H_{12} = \frac{F_2}{F_1} \tag{3.51}$$

Where F_1 is the FFT of one frame of the time signal recorded at mic 1, and F_2 is the FFT recorded at mic 2.

$$H_{12} = \frac{H_{s2}}{H_{s1}} \tag{3.52}$$

where H_{s1} is the transfer function from the generator to mic 1 and H_{s2} is the transfer function from the generator to mic 2. However, following the standard, H_{s1} and H_{s2} should be found from a Fourier transform measured with frequency-domain convolution, followed by an FFT, or by using the complex transfer functions when using an MLS. As with most signal processing, arbitrarily fine resolution can be achieved by zero-padding the impulse responses, after they have been truncated to reduce the effects of background noise.

With the impulse response H_{12} established, the reflection coefficient can be found using:

$$r = |r| * e^{j*\phi*r} = r_r + j * r_c = \frac{H_{12} - H_I}{H_R - H_{12}} * e^{2*j*k_0*x_1} \tag{3.53}$$

where r_r is the real component, x_1 is the distance between the reference plane of the sample and the further microphone location, and ϕ_r is the phase angle of the normal incidence reflection coefficient. The complex wave number k_0, the transfer function for the reflected wave H_R, and for the incident wave H_I, requires a larger introduction. Both transfer functions are reliant on k_0, making it obvious to discuss first.

The complex wave number is <u>not</u> described in DS/ISO 10534-2-2023. It is, however, described in DS/ISO 10534-1-2001 to determine the tube attenuation. k_0 is given by equation 3.54, where k_0' is given by equation 3.55 and k_0'' is given by equation 3.56.

$$k_0 = k_0' - j * k_0'' \tag{3.54}$$

$$k_0' = \frac{2 * \pi}{\lambda_0} \tag{3.55}$$

$$k_0'' = 1.94 * 10^{-2} * \left(\sqrt{\frac{f}{c_0}} * d \right) \tag{3.56}$$

where λ_0 is the wavelength to the current frequency, f is frequency, and d is diameter for circular tubes, or $\frac{4*(W*H)}{2*W+2*H}$, with H being the height and W the width of a rectangular tube. In the standard, d for rectangular tubes is worded as: "*... the ratio of four times the cross-sectional area to the perimeter for a rectangular tube.*", [13]. It should however, be noted that the estimated k_0'' assumes all surfaces within the tube to be completely reflectant, and must therefore be considered the lower limit of how much the tube attenuates. To arrive at the exact value of k_0'', several measurements must be made using the standing wave method for each desired frequency, [13].

With k_0 established, the transfer functions for incident and reflected waves can be given as:

$$H_I = \frac{P_{2I}}{P_{1I}} = e^{-j*k_0*(x_1-x_2)} = e^{-j*k_0*s} \tag{3.57}$$

$$H_R = \frac{P_{2R}}{P_{1R}} = e^{j*k_0*(x_1-x_2)} = e^{j*k_0*s} \tag{3.58}$$

Where P_I and P_R are the pressure magnitudes at the reference plane ($x = 0$). Furthermore, pressure is measured at microphone positions x_1 and x_2, yielding the full picture as shown in Annex C of DS/ISO 2023-2-2023, [14]. Equations to obtain P_I and P_R are left out, as k_0 will be used instead.

As known from the problem analysis, the absorption coefficient can be found, now that the reflection coefficient is determined. This is done by equation 3.59, where r_r is the real component of the reflection coefficient, and r_i the imaginary.

$$\alpha = a - |r|^2 = 1 - r_r^2 - r_i^2 \tag{3.59}$$

As the SWT, the MMM is also capable of determining impedance and admittance properties of the sample. This is done by equations 3.60 and 3.61, where R and g are real components, and X and b are imaginary. As always, $\rho_0 * c_0$ is the characteristic impedance of air.

$$\frac{Z}{\rho_0 * c_0} = \frac{R}{\rho_0 * c_0} + j * \frac{X}{\rho_0 * c_0} = \frac{1+r}{1-r} \tag{3.60}$$

$$G * \rho_0 * c_0 = g * \rho_0 * c_0 - j * b * \rho_0 * c_0 = \frac{\rho_0 * c_0}{Z} \tag{3.61}$$

3.3.3 Comparison to the Standing Wave Method

The MMM offers quite a few improvements compared to the SWT. The largest improvement is the possibility of measuring broadband signals in a single instance rather than having to measure individual frequencies one at a time. Moreover, the MMM has a wider frequency band available, as it does not rely on the formation of standing waves. While the SWT required substantial note-taking during measurements, the MMM requires far less. However, the transfer functions has to be calculated for all materials when utilizing the MMM, leading to a larger computational load.

The list could be longer, but these are the three main differences between the SWT and MMM. Apart from the this, the MMM can also be used for transmission coefficient measurements, which will not be discussed in this project, unfortunately.

3.4 Signal Processing

To gain any knowledge about what is happening within the impedance tube, some level of signal processing is necessary. To create a full perspective of significant aspects, the signal processing will include knowledge of the signal generation, as this enables further optimization of the complete system. The idea is to use fixed-point arithmetic for all computations, maximum length sequence (and/or chirps) for signal generation, and FFTs to analyze the acoustic field within the tube.

3.4.1 Fixed Point Arithmetic

To increase the computational efficiency of the signal processing algorithm (SPA), fixed-point arithmetic (FPA) is used. The benefit of using FPA is that inherently fewer operations are necessary to conduct common operations, such as $+, -, *, /$, unless the hardware is optimized for floating point arithmetics. When binary FPA is used for adding and subtracting, it can be done in a single clock cycle. Furthermore, when dividing or multiplying by 2, bitshifting enables such operands to happen in a single cycle as well. Compared with floating point arithmetic, FPA represents all numbers as scalable integers. When representing numbers in FPA, the bit value is divided into a fractional and an integer part. The major flaw of FPA is acquiring the optimal ratio of which bits should represent the fractional part, and how many bits should represent the integer. By determining the optimal ratio, it is meant to determine how many bits are used to describe each value.

If a 16-bit number is taken as an example, it represents $2^{16} = 65536$ different values in total. However, equally dividing it into 8 bits for the integer and 8 bits for the fraction leaves 256 different values for each, which, depending on the purpose, can be appropriate or flawed. Dividing bits in this manner can be described by Q-formatting, defined by Texas Instruments as "Qm.n", where m is the number of integer bits, and n is number of fractional bits, [47]. By taking 1 bit from the integer part and adding it to the fraction, the ratio shifts to 128 and 512 values. In this example, the integer is now too small to function in some audio applications, as sound easily can surpass 128dB, BUT, if the application regards assessing background noise at high precision, it might still be valid. In general, a fixed-point number's bit ratio can be described as:

$$2^b \cdot \frac{1}{2^{B-b}} \tag{3.62}$$

Where b is the number of bits used for the integer, and B is the total number of bits available. The $\frac{1}{2^{B-b}}$ is meant to show the resolution achievable with the remaining bits. By inserting the desired maximum value or minimum resolution, it is easy to conclude whether or not the current bit-ratio is correct.

However, all is not perfect in the land of FPA, as the first bit (MSB) often signifies a sign, thus reducing the amount of bits. Furthermore, when choosing bit sizes, an overhead should be added to the requirements as well, to minimize the possibility of bit overflow, rendering data flawed. As mentioned previously, the available amount of bits can also lead to insufficient precision in the fractional part, as the number is a fraction and therefore discrete, rather than continuous.

Fortunately, FPA provides many benefits as well, such as predictable execution time, reduced hardware complexity, increased usability for real-time applications, and, when used correctly, complete control of how data is handled across both software and hardware, creating a sturdier system.

Within audio and acoustic signal processing, the use of FPA is highly beneficial, as FFTs, IIR/FIR filters, etc., all utilize a series of additions and multiplications. To see how FPA is used and the code behind it, it is better to visit section 5 on page 51 or the accompanying GitHub with code.

3.4.2 Maximum Length Sequence - MLS

Maximum length sequences (MLS) are a type of pseudorandom binary sequence, which can be generated by a linear feedback shift register (LFSR), such as the one shown in figure 3.8. The main benefit of MLS is that, though the sequence mimics random noise, it is deterministically produced. The name comes from the sequence using all nonzero binary values in an LFSR, thus creating the maximum length sequence. The largest possible length is given by $2^n - 1$, where n is the number of stages in the register, and the -1 removes the possibility of hitting an all-zero stage.

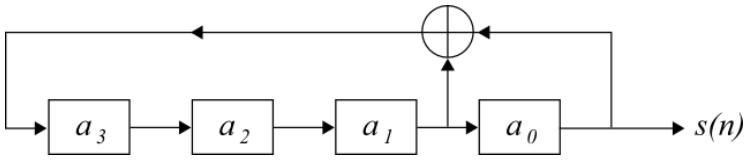


Figure 3.8: Linear feedback shift register of length 4, [48].

MLS are generated by the shift register from a given seed to ensure that sequences are reproducible. As the shift register cycles through all possible values, the register seed can be any value except for the all-zero state. The recursive function for figure 3.8 is described in equation 3.63, where it can be seen to be periodic.

$$\begin{cases} a_3[n \oplus 1] = a_0 \oplus a_1[n] \\ a_2[n \oplus 1] = a_3[n] \\ a_1[n \oplus 1] = a_2[n] \\ a_0[n \oplus 1] = a_1[n] \end{cases} \tag{3.63}$$

Where n is the time index and $\oplus$ is the modulo-2 addition, equaling the XOR operation. When implemented, the shift register can create sequences based on its length N,

by 2^N. Therefore, an LFSR with 16 flip-flops can create a sequence of $65536 - 1$ samples, with the -1 sample being all-zero.

When examining a period, the bitstream can be divided into runs. A run is defined by a subsequence of identical symbols within a period, with the length of this subsequence being the length of the run, [49]. For all periods/sequences, the following properties should be met:

- The number of ones equals the number of zeros plus 1
- Each period must contain a run of length N
- One half of the runs are of length 1
- One quarter of the runs are of length 2
- One eighth of the runs are of length 3
- One-sixteenth of the runs are of length 4
- One-sixtyfourth of the runs are of length 5

When creating the LFSR, XOR gates are put at specific "taps" to yield the MLS. The taps are best found by inspecting a table, such as G.1 found in the appendix on page 126, as manually finding them can be tedious. However, they can be found as coefficients to a primitive polynomial on GF(2)[1], which can be calculated using Eulers totient function, [48, 50]. The tap's main function is to specify which flip-flop outputs should be XOR'ed together. In equation 3.64, a 4-bit feedback polynomial can be seen, with coefficients at x^4 and x^3. This means that the taps are at the third and fourth bit positions, and they should be XOR'ed together to yield the correct feedback. When XOR'ing the bits together, a maximum length sequence for 4 bits is given in table 3.2, where the "output bit" column yields the MLS with this specific seed. If a different set of flip-flops is used to generate the sequence, it is not certain that it will be an MLS, [51, 48, 49, 52].

$$x^4 + x^3 + 1 \tag{3.64}$$

In table 3.2, it can be seen that the LFSR is periodic within 15 cycles, matching the $2^N - 1$ specification. Furthermore, with the taps correctly placed, it will never yield an all-zero value, which can also be seen, as it is the only combination left out in the table. The output MLS is "100110101111000", which can be divided into the following runs: 1 | 00 | 11 | 0 | 1 | 0 | 1111 | 000. This matches the properties mentioned earlier, as the number of zeros is 1 less than ones, half the runs are of length 1, a quarter is of length 2, an eighth is of length 3, and the first flaw is found, as more than one sixteenth is of length 4. However, this meets the criteria of a run being the same length as N.

As the amount of bits N expands, the period expands by twos complement, which can also be seen in the period column of table G.1 on page 126. The properties given by a larger number of bits yield a significantly larger period, which on the surface will seem even more random than the 4-bit example, while still being 100% deterministic.

[1]Galois finite field: a finite set of numbers with defined addition, subtraction, multiplication, and division operations, where every nonzero element has a multiplicative inverse.

Iteration	Start Value	XOR Result	New Value	Output Bit
1	1001	1	0011	1
2	1100	0	0110	0
3	0110	1	1100	0
4	1011	1	0101	1
5	0101	1	1010	1
6	1010	1	1101	0
7	1101	1	1110	1
8	1110	0	1111	0
9	1111	0	0111	1
10	0111	0	0011	1
11	0011	0	0001	1
12	0001	1	1000	1
13	1000	0	0100	0
14	0100	0	0010	0
15	0010	1	1001	0
16 (1)	1001	1	1100	1

Table 3.2: Table of XOR operations with taps at the third and fourth bit. The output bits are the values in the MLS, meaning that the corresponding MLS for a 4-bit LFSR with this seed (1001) will cycle these 15 values indefinitely, [51].

Autocorrelation Properties of MLS

One of the many useful features of MLS is the autocorrelation property. For any binary sequence $s(t)$ of period $2^N - 1$, the autocorrelation function at a time is defined as:

$$R(\tau) = \sum_{t=0}^{2^N-1-1} (-1)^{s(t)\oplus s(t+\tau)} \tag{3.65}$$

wheres $s(t)$ is the bit at time t and $\oplus$ is modulo-2 addition (XOR). The sequence is often portrayed with $0 = -1$ and $1 = +1$ for the respective bit values, when doing correlation analysis. At zero lag ($\tau = 0$), the sequence isn't shifted, resulting in bits lining up perfectly, and a perfect correlation $R(0) = 2^N - 1$. However, for all other shifts/$\tau \neq 0$ the correlation will be -1, making it seemingly random. The goal of autocorrelation is to check whether or not the shifted sequence resembles the non-shifted sequence. The constant mismatch results in a correlation of -1, matching total randomness.

The correlation itself is an important measure as it in practice can act as an identifier when measuring an impulse response in a noisy environment. As the correlation is 1 at only one spot, the impulse response can be swept through until found, and thereafter the original signal can be omitted from the response.

Creating White Noise from MLS

The MLS is highly beneficial to use in acoustic system identification, as the spectral composition of an MLS is nearly identical to the randomness of white noise, [53, 54, 50, 55]. When playing an MLS through a speaker, it can be seen as a linear time-invariant (LTI) system. The bipolar signal of MLS (-1 to +1) corresponds to the minimum and maximum positions of the speaker cone, and the constant switching creates a vaguely sinusoidal signal of different wavelengths. While the speaker is not playing exact white noise, the MLS is near enough for human hearing to consider it white noise. Furthermore, as the Fourier transform of an MLS approximates a flat spectrum, all frequencies

are played equally, making the MLS signal indistinguishable from white noise. Mathematically, the LTI system can be described by equation 3.66, and the Fourier transform is given by equation 3.67, [53, 55].

$$y[n] = x[n] * h[n] = \sum_{k=0}^{N-1} x[k] * h[n-k] \tag{3.66}$$

$$X(f) = \sum_{n=0}^{N-1} x[n] * e^{-j*\frac{2*\pi*f*n}{N}} \tag{3.67}$$

Where $x[n]$ is the MLS signal and $h[n]$ is the system response. If the example used in [53] is used instead, the MLS used in regards to room acoustics can be given by equation 3.68, and the system as a whole can be seen in figure 3.9. The general idea is to play the MLS on an LTI system (speaker), creating an impulse response of a room. Thereafter, a fast Hadamard transform can be used to derive the room's impulse response. From that impulse response, a band filter can reveal the decay curve for specific octave bands[2], and an FFT will reveal the spectrum. The main benefit from utilizing this method is that the SNR can be increased by 3 dB each time the number of averages obtained is doubled, effectively enabling a noise-free impulse response with enough averages available, [53, 55].

$$s(t) = m(t) * h(t) \leftrightarrow s[n] = m[n] * h[n] \tag{3.68}$$

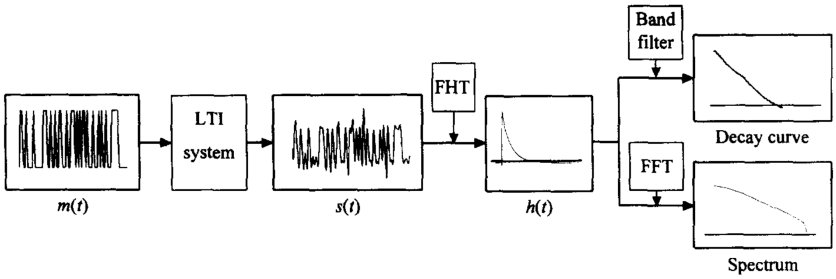


Figure 3.9: Signal flow schematic of MLS measurement, [53].

3.4.3 Chirps

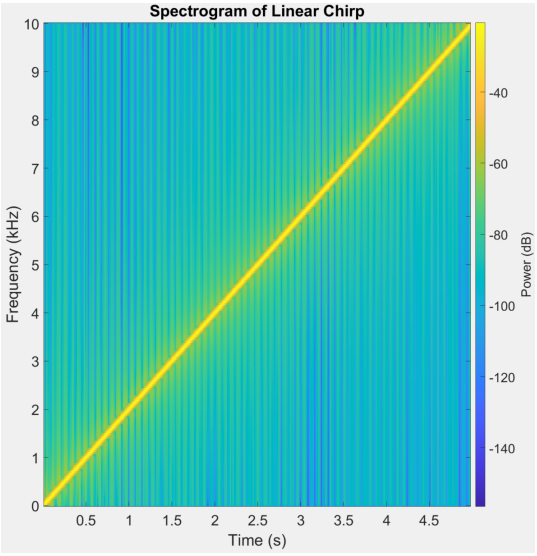
Chirps are a common method of gaining impulse responses over a broad range of frequencies, [56, 57, 58, 59]. A chirp is a form of frequency sweep that will either increase in frequency (upsweep) or decrease (downsweep). Several types of chirp exist, but only the linear and exponential types will be described in this section. Mathematically, the linear chirp can be described by equation 3.69 and the exponential chirp by equation 3.70. Spectrograms of both types can be seen in figure 3.10, [56, 57, 58, 59].

$$x(t) = A * cos(2 * \pi(f_0 t + \frac{k}{2} t^2) + \phi_0) \tag{3.69}$$

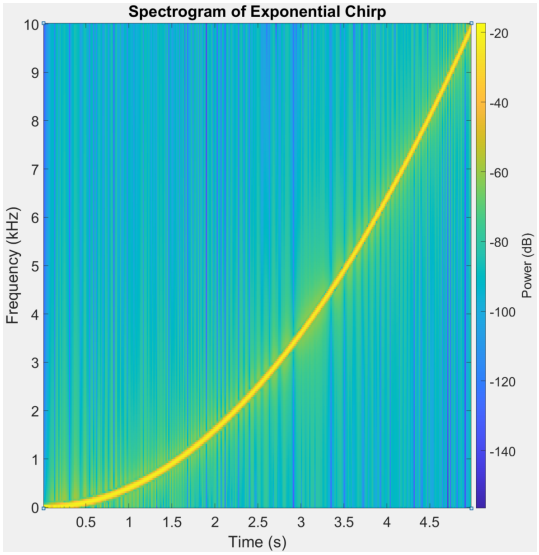
$$x(t) = A * cos(2 * \pi * f_0 \left(\frac{b^t - 1}{ln(b)}\right) + \phi_0) \tag{3.70}$$

Where A is the amplitude, f_0 is the initial frequency, $k = \frac{f_1 - f_0}{T}$ is the "chirp rate", f_1 is the final frequency at time T, ϕ_0 is the initial phase, and b is a scaling factor for the exponential growth of the frequency given by $b = \frac{f_1}{f_0}^{\frac{1}{T}}$.

[2]For information about octaves, see appendix E on page 122.



(a) Spectrogram of a linear chirp, generated with MATLAB's chirp function.



(b) Spectrogram of an exponential chirp, generated with MATLAB's chirp function.

Figure 3.10

In general, chirps are useful in system identification, as each chirp can be modified to fit broad frequency bands or smaller octave bands, fitting the application at hand. Compared with white noise and MLS, a chirp provides equal energy across the frequency spectrum, but the main advantage is a higher SNR. For MLS and white noise, the response has to be averaged across multiple tests, whereas with chirps, matched filters allow higher SNR, [56, 57, 58, 59].

4 Demand Specification

4.1 High Level Specification

This section describes a high-level overview of what the impedance tube should be capable of. The functionality is written as seen by an acoustical engineer interested in a new impedance tube focused on low frequencies. Detailed specifications can be found in section 4.2 starting page 49.

As an acoustical engineer with a focus on low frequencies, I want:

- *An impedance tube that can measure beneath the current state-of-the-art impedance tubes.*
- *An impedance tube capable of fitting square samples.*
- *An impedance tube capable of fitting large samples.*
- *An impedance tube wherein perforated materials can be tested.*
- *An impedance tube outputting .CSV data for data handling.*
- *An impedance tube with an intuitive interface.*
- *An impedance tube where I can inspect whether my sample is placed correctly in the tube.*
- *An impedance tube where I can easily switch signal output.*
- *An impedance tube wherein deep samples can be placed to test low frequency absorption.*
- *An impedance tube that auto-calibrates microphones.*
- *An impedance tube which calculates the impedance and admittance of a material.*
- *An impedance tube that complies with DS/ISO 10534-2-2023.*
- *An impedance tube that complies with ASTM 2611-24.*

4.2 Functional Specification

This section describes the functional criteria of the product. The criteria are seen from an end-user perspective and made as user stories, where acceptance criteria (AC1 & AC2, e.g.) must be fulfilled. A test of the functional specifications is made in section 6.

4.2.1 Measuring Infrasound

As an acoustical engineer, I want an impedance tube that can measure beneath the current state-of-the-art impedance tubes.
Accept Criteria:
AC1: The impedance tube must be able to measure below 35 Hz (currently the lowest measurable frequency, [60]).
AC2: The impedance tube should be able to measure below 20 Hz.
AC3: The impedance tube should be able to measure below 10 Hz.
AC4: The impedance tube should be able to measure down to 1 Hz.

4.2.2 Fitting Square Samples

As an acoustical engineer, I want an impedance tube capable of fitting square samples.
Accept Criteria:
AC1: The tube style must be square.

4.2.3 Fitting Large Samples

As an acoustical engineer, I want an impedance tube capable of fitting large samples.
Accept Criteria:
AC1: The tube should, at a minimum, be capable of fitting 10x10cm samples.
AC2: Adapters should enable the tube to fit samples up to 30x30cm.

4.2.4 Data Output in CSV Format

As an acoustical engineer, I want an impedance tube outputting .CSV data for data handling.
Accept Criteria:
AC1: Data must be saved or transmitted to local storage in .CSV format.

4.2.5 Intuitive Interface

As an acoustical engineer, I want an impedance tube with an intuitive interface.
Accept Criteria:
AC1: The interface should be accessible at the impedance tube.
AC2: Menus, data, and interpretations must be easily read.
AC3: Results must be easy to extrapolate.
AC4: Results, measurements, etc. should be saveable to external storage.

4.2.6 Correct Sample Placement Inspection

As an acoustical engineer, I want an impedance tube where I can inspect whether my sample is placed correctly in the tube.
Accept Criteria:
AC1: The tube should be see-through.
AC2: Samples can be placed in a jig out of the tube, which can be placed within the tube.

4.2.7 Easy Signal Output Switching

As an acoustical engineer, I want an impedance tube where I can easily switch signal output.
Accept Criteria:
AC1: The speaker must be capable of playing an MLS signal.
AC2: The speaker must be capable of playing white noise.
AC3: The speaker must be capable of playing a chirp or similar frequency sweep.
AC4: The speaker must be capable of playing a pure sinusoid.

4.2.8 Testing Deep Samples for Low Frequency Absorption

As an acoustical engineer, I want an impedance tube wherein deep samples can be placed to test low-frequency absorption.
Accept Criteria:
AC1: The sample holder must be adjustable from 2-10 cm.
AC2: An extra sample holder adjustable from 10-50 cm must be available.
AC3: An extra sample holder adjustable from 50-100 cm must be available.

4.2.9 Auto-Calibration of Microphones

As an acoustical engineer, I want an impedance tube that auto-calibrates microphones.
Accept Criteria:
AC1: Instead of switching microphone positions for calibration, the algorithm must calculate corrected transfer functions based on microphone position, temperature, humidity, and adequate testing signals.
AC2: The calibration must adhere to DS/ISO and ASTM standards.

4.2.10 Impedance and Admittance Calculation

As an acoustical engineer, I want an impedance tube which calculates the impedance and admittance of a material.
Accept Criteria:
AC1: The calculated impedance must match the Smith chart method.
AC2: The calculated admittance must match the Smith chart method.
AC3: The real and imaginary values must be outputted as well.

4.2.11 Compliance with DS/ISO 10534-2-2023

As an acoustical engineer, I want an impedance tube that complies with DS/ISO 10534-2-2023.
Accept Criteria:
AC1: Must comply with the relevant criteria set by DS/ISO 10534-2-2023.

4.2.12 Compliance with ASTM 2611-24

As an acoustical engineer, I want an impedance tube that complies with ASTM 2611-24.
Accept Criteria:
AC1: Must comply with the relevant criteria set by ASTM 2611-24.

5 System Design

The system-design chapter will be structured into 4 sections: "Physical Setup", "Creating Sound", "Measuring Sound", and "Establishing an Algorithm For Data Handling". The sections will dive into what it takes to create each aspect of the system, along with their respective implementations and system-specific tests.

In figure 5.1, an overview has been made of the system. The "tube" section encompasses all the physical aspects of the impedance tube, while the "microcontroller" (MCU) section divides all of the tasks that should run hereon.

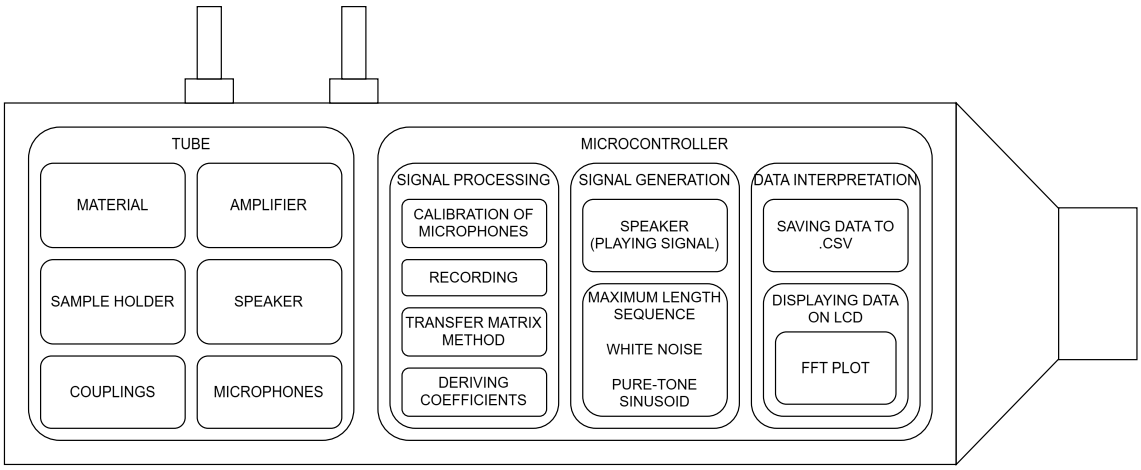


Figure 5.1: Overview of the system-design chapters, and what aspects they will contain.

5.1 Physical Setup

The physical part of the impedance tube will consist of the tube itself, a speaker cabinet, microphones, and a sample holder. Each component will be examined and have characteristics determined by using the DS/ISO 10534-2-2023 standard. In figure 5.2, a first draft is visible. The tube has been chosen to be square, to fit the functional specification.

The tube has been angled to fit the speaker cabinet used for the standing wave apparatus type 4002 by Brüel & Kjær, as a way to ensure the tube's inherent characteristics are correct. This is imagined to be done by placing a sample with known absorption and reflection in both the standing wave tube and the new impedance tube, and then comparing the results. The braces/feet are meant to help stiffen the tube while holding it at the correct height to fit the 4002 speaker. The red cylinders are the minimum distances used for microphone placement, while blue is adhering to the 250mm recommendation, and green are the apparent optimal position, as described in more detail in section 5.1.1.

The tube has been drawn using two layers of 8mm acrylic sheets to gain the most mass, while creating a stiffer structure. This is elaborated in section 5.1.2.

The sample holder is thought to slide over the tube end, creating a uniform connection. This connection type also encourages multiple different sample holders while enabling the user to custom-make holders fit for their specific sample. To obtain a reflective stop at the sample holder, an aluminum block (not visualized) is placed behind the sample, thus making the holder variable depth. To secure the lid, additional braces/feet will be placed, forcing the lid tightly to the box. The sample holder and general ideas behind it will be elaborated in section 5.1.3.

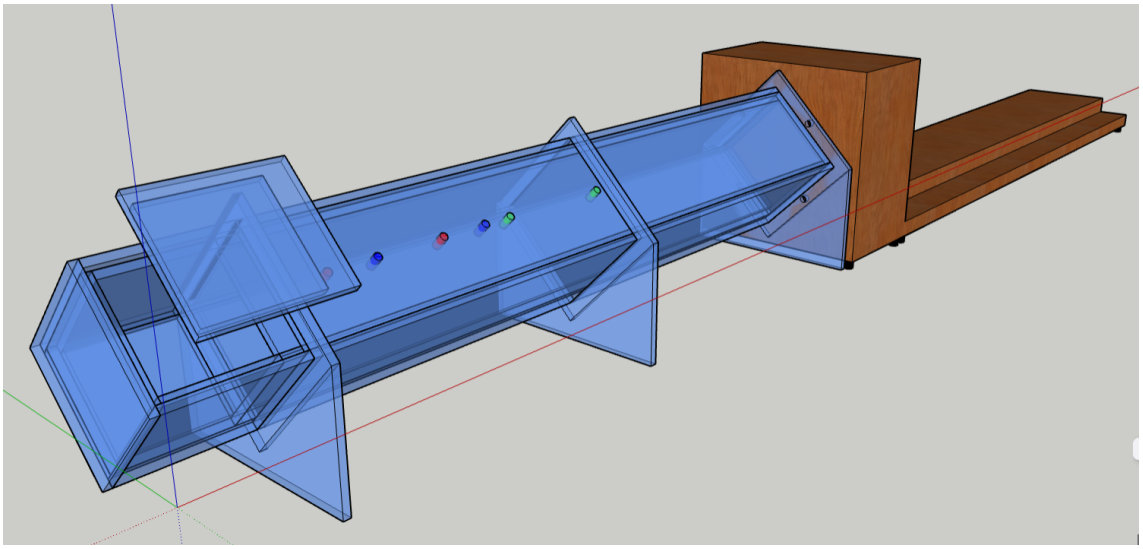


Figure 5.2: Rough draft of what the physical setup could look like. In this sketch, dimensions are given by section 5.1.1 and table 5.1.

5.1.1 Determining Desired Physical Dimensions

The physical dimensions (within the tube) are given by the equations from section 3.3.1 on page 37, and otherwise by DS/ISO 10534-2:2023, [14]. Adhering to the formulas requires an upper desired frequency, however, they can be modified to use desired dimensions instead to determine an upper frequency. As specified in 4.2.3, the minimum sample size should be 10x10cm, yielding a maximum length d across the diagonal of $141.42 * 10^{-3}[m]$. The upper frequency can then be found to be:

$$141.42 * 10^{-3} \leq 0.5 * \lambda_U \rightarrow \lambda_U \geq 282.84 * 10^{-3}[m] \tag{5.1}$$

$$\lambda_U = \frac{c}{f_U} \rightarrow f_U \leq 1212.69[Hz] \tag{5.2}$$

If the desire for fitting 30x30cm samples is used instead, the corresponding upper frequency bound is $404.23Hz$. With the upper frequency established, the distance between microphones s can be set as well:

$$f_U * s \leq 0.45 * c \rightarrow s \leq 127.28 * 10^{-3}[m] \tag{5.3}$$

However, it is now obvious that not all recommendations from the standard can be followed. The standard stated that the microphone furthest from the sample (mic 2) should be at most 250mm, and the nearest microphone (mic 1) should be at least $2 * d$ from the sample. However, as the focus of this specific impedance tube will be on lower frequencies, the distance s between the samples should be as large as possible to yield

optimal results. As can be seen from equation 5.3, the largest viable distance between the microphones is $127.28 * 10^{-3}[m]$, and since the largest dimension d is $141.42 * 10^{-3}[m]$, this places the second microphone $410.12 * 10^{-3}[m]$ from the sample. In table 5.1, different microphone placements can be seen, along with the recommendations that they meet from the standard. Common for all three is that they meet the recommended distance between microphones for optimal low frequency analysis and the recommended distance of $3 * d$ between mic 2 and the speaker.

Type	Distance From Sample Mic 1	Distance From Sample Mic 2	Minimum Tube Length	Recommendations Met
$\frac{1}{2} * d$	$70.71 * 10^{-3}$	$197.99 * 10^{-3}$	$622.25 * 10^{-3}$	Minimum distance from sample to mic 1
$2 * d$	$282.84 * 10^{-3}$	$410.12 * 10^{-3}$	$834.39 * 10^{-3}$	Recommended distance from sample to mic 1
$250 * 10^{-3}[m]$	$122.72 * 10^{-3}$	$250 * 10^{-3}$	$674.26 * 10^{-3}$	Recommended distance from sample to mic 2

Table 5.1: Tube lengths and microphone distances according to different recommendations given by DS/ISO 10534-2-2023.

As the recommended distance from the speaker to mic 2 is the minimum distance, it is possible to create a tube, which could be used for each configuration. If the tube length is chosen as the longest possible, then the shorter versions can be placed as well. This can be done by creating 6 microphone slots in the tube, which can either be used in pairs, or for complete analysis, all be used at once. Furthermore, a version with more than 2 microphone slots would enable the user to do measurements aimed at higher frequencies, as microphones can be placed closer for better resolution at these frequencies. Optimally, a large number of microphone slots can be made 70 to $410 * 10^{-3}[m]$ from the sample, to enable customizable measurements. For now, the focus will be to arrange 6 microphones in the longest possible tube.

With that, the tubes (inner) dimensions are $L = 834.89 * 10^{-3}$, $W = 100 * 10^{-3}$, $H = 100 * 10^{-3}$, $d = 141.42 * 10^{-3}$, with microphones placed according to table 5.1. Apart from the tube itself, a sample holder and speaker cabinet should also be added to the total dimension. The sample holder is discussed in section 5.1.3 on page 54 and the speaker cabinet in 5.1.4 on page 55.

5.1.2 Choosing Adequate Tube Materials

Following the standards, sufficiently hard, rigid, heavy, and smooth materials should be used for the tube. However, specific values are not given per se. Instead, a steel tube is exemplified as being adequate if the wall thickness is 5% of the diameter d or 10% of the diagonal dimension for rectangular tubes. Furthermore, concrete tube solutions are mentioned, without any examples of dimensions. The tubes should in all cases be "*heavy and thick enough not to be excited to vibration by the sound signal and not to show vibration resonances in the working frequency range of the tube.*", [13, 14]. In table 5.2, some examples of different available materials can be seen, along with their mass.

Considering these values, it is possible to calculate how thick a sheet of concrete/-plywood/acrylic/glass would have to be to match that of steel. Following the recommendation of a steel wall being 10% of the diagonal dimension, it results in a thickness of $14.14 * 10^{-3}[m]$. Using this thickness, each cm^2 would weigh $11.1g$. To achieve an

identical weight for each cm^2, the thickness of relevant materials can be seen in table 5.2. As these thicknesses are significantly higher than what is feasible in reality, a different approach is made.

Material	Density	Thickness to match steel density
Steel, [61]	$7.85\left[\frac{g}{cm^3}\right]$	$14.14 * 10^{-3}[m]$
Concrete, [62]	$2.4\left[\frac{g}{cm^3}\right]$	$46.26 * 10^{-3}[m]$
Plywood, [63]	$0.54\left[\frac{g}{cm^3}\right]$	$205.58 * 10^{-3}[m]$
Acrylic, [64]	$1.18\left[\frac{g}{cm^3}\right]$	$94.08 * 10^{-3}[m]$
Glass, [65]	$2.5\left[\frac{g}{cm^3}\right]$	$44.41 * 10^{-3}[m]$

Table 5.2: Approximate mass densities of various materials which could be used in an impedance tube and their required thickness to match the weight of steel.

If the functional specification "Correct Sample Placement Inspection" is revisited, the desire to view the samples during testing should be considered as well. With that in mind, the possible materials are glass or acrylic. Due to the intricacies of working with glass, it will also be disregarded as a possible solution, even though it would hit the "heavy" requirement far better than acrylic.

Instead of focusing on weight, a more subjective aspect is chosen, with hold in the materials available at AAU. Two layers of acrylic will compose the structure of the tube. However, this allows the use of the laser cutter to produce components. Furthermore, the acrylic supplies a sufficiently smooth surface, along with rigidity conforming to the 0.2% deviation demand set by the standard. To achieve the desired stiffness/deadness, braces are placed along the tube, which can be seen in figure 5.2. Unfortunately, the height and width of the impedance tube will be too large to fit within the bolts at the Type 4002 speaker cabinet if two layers of 8mm acrylic are used. Therefore, the inner tube will be 8mm thick to create sufficient stiffness, and the outer tube will be 6mm to add as much weight as possible.

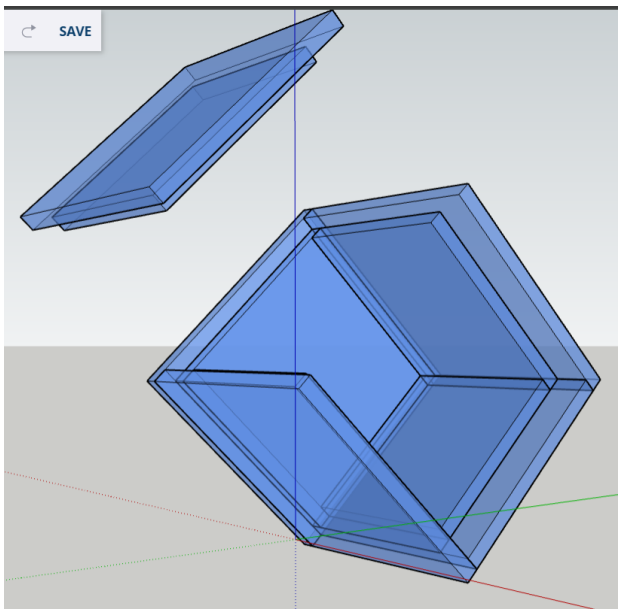
NOTE: The available acrylic sheets at AAU have dimensions of $80 * 50 * 10^{-3}[m]$*, not making it possible to achieve the desired length of* $834 * 10^{-3}[m]$ *in a single sheet. Therefore, an additional coupling will be made, dividing the impedance tube into three total parts: 1 length from the speaker to the microphones (blank tube), 1 part containing the microphone array (microphone tube), and 1 part containing the sample (sample holder).*

5.1.3 Creating a Sample Holder

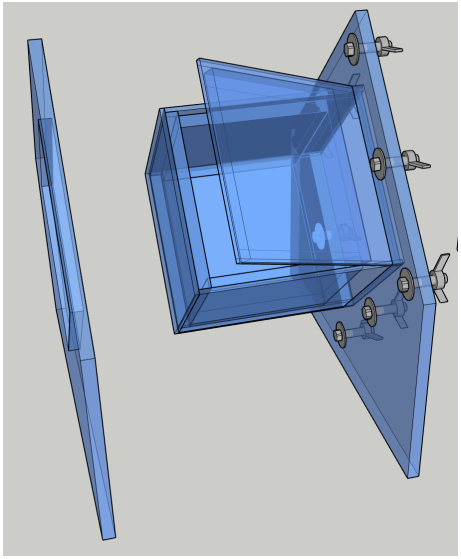
The sample holder will be made out of 8mm laser-cut acrylic sheets as well. To handle the hassle of shoving a sample into a tube, the sample holder should be made as a square "bowl", with one side removable as a lid. This can be seen in detail in figure 5.3a.

As mentioned briefly in the introduction to the system-design chapter, the idea of creating sample holders like this is to make them variable depth, while ensuring ease of sample installation. The sample itself should be placed at the front of the sample holder, flush with the center tube's edge. Behind the sample, an aluminum block should be placed, creating a sufficiently reflective backing, which can be moved back and forth, according to sample thickness.

Apart from a few standardised sample holders, this simple design allows users to easily produce sample holders of their own that fit specific measurements. That could



(a) Example of what the sample holder could look like.



(b) Improved version of the sample holder, where 1 brace is glued to the holder. The lid and secondary brace is slid into place.

Figure 5.3

be if a company were to develop the best acoustic absorber within 25mm, a sample holder capable of fitting 25 mm-thick samples could be produced within an hour.

Creating a sufficiently good lid mechanism is somewhat complicated. Returning to the first draft, the idea was that the braces were made with little to no tolerance, ensuring that the braces would clamp the lid shut. While a scaled model showed some hassle when assembling as first intended, the design has been modified to slide the lid on before the rearmost brace is placed. This can be seen in figure 5.3b

5.1.4 Speaker Chamber

To utilize the availability of the Brüel & Kjær Type 4002 apparatus, the speaker chamber will be taken from it and attached to the impedance tube. By utilizing the speaker chamber from the standing wave apparatus it enables the possibility of cross-referencing results with results obtained using the standing wave method.

If a speaker chamber had to be made from scratch, several considerations should be made. First and foremost, the speaker itself should be mechanically disconnected from the box itself, to debilitate the possibility of vibrations being transferred into the tube. This can be done by suspending the speaker in springs or similar, creating a "floating" speaker. Secondly, the cabinet/box/chamber/enclosure itself must not transfer reflections from the cabinet to the tube, to ensure that only the reflective/absorptive coefficients of the sample itself are measured, [13, 14]. Effectively, this means that the enclosure should be anechoic.

Furthermore, the speaker itself must occupy at least $\frac{2}{3}$ of the tube's cross-sectional area according to both versions of the 10534 standard, [13, 14].

5.1.5 Couplings Between Elements

To assemble the tube with sample holders, itself, and the speaker cabinet, couplings are needed. The couplings are integrated into the braces, stiffening the tube, by adding holes through which bolts shall go. In each brace, 12 6mm holes fit for an M6 bolt shall

be made. Each bolt will be fitted with a 20mm washer for even pressure distribution, and a wingnut is used to ease installation. In each coupling, the outer tube is retracted 25mm on one part and extended 24mm on the other. This 1mm difference is important, as it will ensure that the inner tubes are completely against each other when the braces are bolted together. The extended/retracted design is chosen to ensure that inner tubes are completely aligned, conforming to the DS and ASTM standard of deviation within the tube.

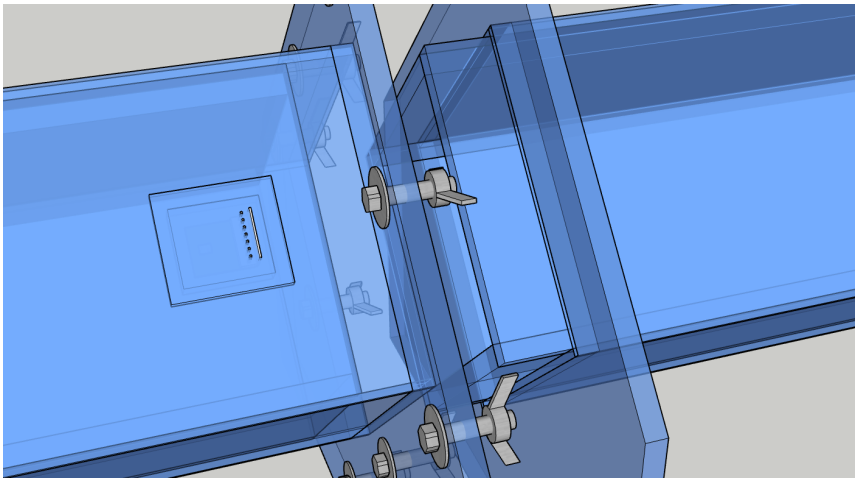


Figure 5.4: Coupling between each impedance tube element, showcasing bolts, washers, wingnuts, and the retraction/extension of each tube.

5.1.6 Microphones and Their Placement

As mentioned in section 5.1.1, a total of 6 microphones will be placed in the tube. The microphones are placed away from the sample at:

- $mic_{11} = 70.71 * 10^{-3}[m]$
- $mic_{12} = 197.99 * 10^{-3}[m]$
- $mic_{21} = 282.84 * 10^{-3}[m]$
- $mic_{22} = 410.12 * 10^{-3}[m]$
- $mic_{31} = 122.72 * 10^{-3}[m]$
- $mic_{32} = 250.00 * 10^{-3}[m]$

To enable more microphones to be used in the future, sockets will be embossed into the side of the impedance tube, as shown in figure 5.5. These sockets will be interchangeable, allowing easy augmenting/switching of microphones. For this project, PUI Audio DMM-4026-B-I2S-R MEMS microphones will be used, and the sockets made will therefore fit these. If different microphones should be used, a new socket should be made, encompassing the new microphone.

The PUI Audio DMM-4026-B-I2S-R microphones have been chosen as they're fairly cheap, can interface with an MCU with I2S, and are available from the component selection at AAU. Furthermore, utilizing MEMS microphones enables the creation of an even more approachable system as a whole. The microphones has a frequency range of 20Hz-20kHz, which isn't optimal in regards to functional specification 4.2.1. However, this is bypassed for now to accommodate the system as a whole instead. As the microphone sockets are made to be interchangeable, more sensitive microphones can be used in future iterations.

Regarding precision, the PUI Audio DMM-4026-B-I2S-R delivers 24-bit data sizes with 18 bits of precision, and a wrapper consisting of 8 bits, totalling a 32-bit word size. The microphones can be set up in mono, stereo, or array configurations. The digital I2S

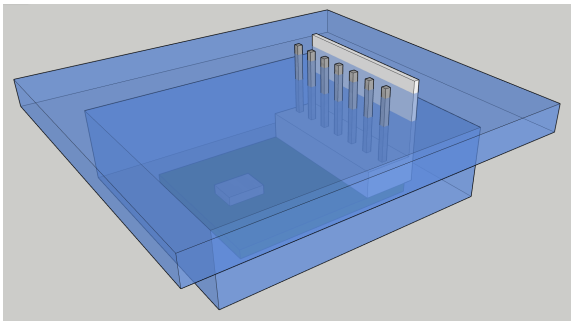


Figure 5.5: Microphone socket with microphone PCB placed within. Negatives of the socket are made in the tube, making installation easy.

interface removes the need for any sort of ADC, preamplification, or similar processing before being input to the MCU. Although similar alternatives were considered, the decision was made to focus on the development of the tube and DSP algorithm, rather than the electrical development of microphone and speaker systems. The G.R.A.S. "40AZ 1/2inc Prepolarized Free-field Microphone, Low Frequency" microphones, along with G.R.A.S. "26CC 1/4inch CCP Standard Preamplifier with SMB Connector" were initially chosen as the desired setup, as it could measure frequencies as low as 0.5Hz. However, this setup was neglected due to the amount of additional components needed before an analog signal could be input to the ADC on an MCU.

5.1.7 Choosing a Microcontroller Unit

At first, the considered microcontroller (MCU) was an ESP32 or similar affordable MCUs. However, an acquaintance suggested the Teensy 4.1 as an option, and after exploring its capabilities, it has been deemed more than sufficient for this project. As listed on the official website, the relevant specs of a Teensy 4.1 are:

- ARM Cortex-M7 at 600 MHz
- Floating point math unit, 64 & 32 bits
- 7936K Flash, 1024K RAM (512K tightly coupled), 4K EEPROM (emulated)
- QSPI memory expansion, locations for 2 extra RAM or Flash chips
- USB device 480 Mbit/sec & USB host 480 Mbit/sec
- 55 digital input/output pins, 35 PWM output pins
- 18 analog input pins
- 8 serial, 3 SPI, 3 I2C ports
- 2 I2S/TDM and 1 S/PDIF digital audio port
- 3 CAN Bus (1 with CAN FD)
- 1 SDIO (4 bit) native SD Card port
- Ethernet 10/100 Mbit with DP83825 PHY
- 32 general-purpose DMA channels
- Cryptographic Acceleration & Random Number Generator
- RTC for date/time

Furthermore, the Teensy 4.1 also benefits from a dedicated audio shield that interfaces directly with the board, enabling audio input and output through a jack. These audio boards/shields are used with the I2S (Inter-IC Sound) protocol, facilitating real-time communication between DACs, ADCs, and the Teensy board. The 600 MHz ARM Cortex M7 processor provides an abundance of power for use in this project, especially

paired with the integrated FFT libraries specifically made for this board and processor. Apart from the single-point FPU available, the Teensy also supports 32-bit fixed-point arithmetic, allowing sufficiently high resolution in all regards within this system. Utilizing the real-time applicability of the Teensy board, it should be possible to create a real-time analysis of major acoustic characteristics, such as impulse response, frequency response, spectral analysis, SPL, etc.

Moreover, the Teensy 4.1 is programmable in Arduino IDE, making implementation less cumbersome than for direct DSP boards. This should make the system as a whole more approachable for future iterations, as the code will be pure C/C++.

5.1.8 Creating the Impedance Tube

With all physical aspects of the impedance tube described and thought through, it can now be assembled in real life. Schematics for every part of the impedance tube can be seen in appendix "Schematics For Construction of the Impedance Tube" on page 127. It should be noted that as the Type 4002 speaker is placed 1cm within the speaker cabinet, this centimeter is withdrawn from the total length of the impedance tube. Physically, each part of the impedance tube is cut from its respective material before being put together. Acrylic parts are glued together using Acrifix after being placed in a jig to ensure right-angled corners are made. Each tube is made separately and slid into/over each other, before being fixed into position. When the position has been double-checked, Acrifix is applied at all edges, effectively fusing the tubes. When the tubes are joined, the coupling towards the Type 4002 speaker can be glued to the end with Acrifix. Then the braces can be slid into position, possibly using a drop of detergent to enable easier sliding. When placed, the braces should be clamped strongly enough not to slide back and forth, at least when the detergent is dry. The sample holder is glued together in a similar manner using a jig to create uniform corners. The microphone sockets are glued together so that the microphones can be placed in the socket with ease, and be firmly secured. When every part has been made, testing of the physical tube can begin. 3D models of the final design can be seen in figures 5.6 and 5.7.

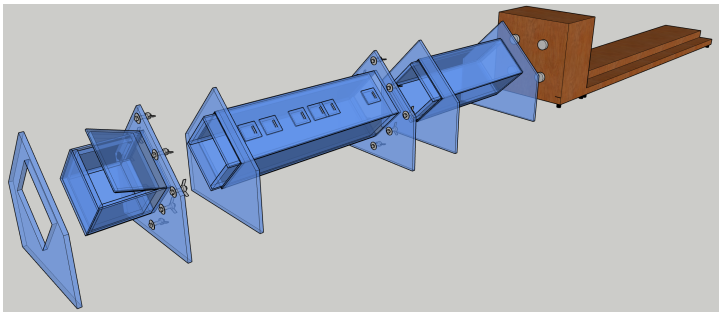


Figure 5.6: Final 3D model of the impedance tube, with all parts open/pulled apart.

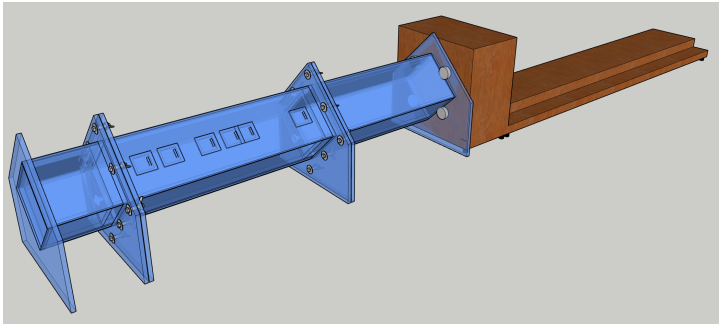


Figure 5.7: Final 3D model of the impedance tube, with all parts closed/pushed together.

Testing the Physical Tube

The tube itself must comply with the following criteria:

- Length, width, and height within 0.2%
- Microphone placements within 0.2%
- Sealable microphone sockets
- Sealable sample holder
- Airtight connection with the Type 4002 speaker
- Clear view of the sample
- Easy adjustment of the terminating end

The tube is put together and then measured/tested to fit within the criteria. A sliding gauge were applicable and otherwise a calibrated HULTAFORS 59-12 yardstick is used to measure the tube sections. However, the sliding gauge has an error of 0.03mm, and the yardstick suffers from human reading, [66, 67]. Nevertheless, measurements made are considered valid. To control the angles, a speed-square is used, and for straightness, a straightedge is used. To determine airtightness, the entire system is put together and filled with pressurized air, and soapy water is applied to all edges, revealing if any leaks are present.

Test results In general the tube has performed as expected. Regarding deviations in measurements, the largest error is 0.39% at an inner measurement in the sample holder, which is above tolerance set for the project. When comparing with the DS ISO 10534:2023 standard, all demands are met, as the maximum allowed deviance is 0.2% in the cross-measure. While a single measurement is at 0.2%, it is still within tolerances. All microphone placements and all measurements beside 2 outer tube and 1 inner tube, are well within tolerance. This can more easily be seen in table 5.3 on page 62.

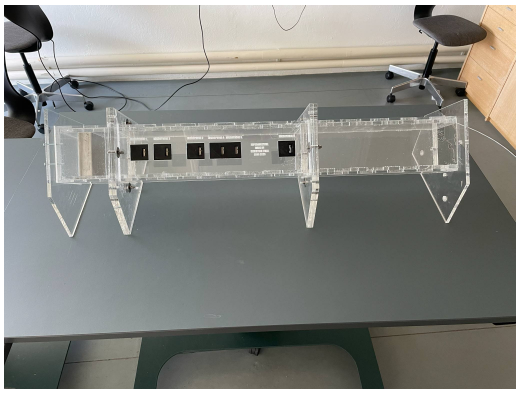
When checking right angles, straightness, etc. no noteworthy deviation has been found. The largest deviation found is in the connection between the blank tube and the microphone tube, where an edge can be felt, but not directly measured. For now, this is considered OK, despite it does not comply 100% with the DS standard.

Regarding air-tightness, the system has been found to be mostly airtight without any petroleum jelly used. Most leaks are present around the microphone sockets, and are expected to be neutralized when petroleum jelly is applied for further testing.

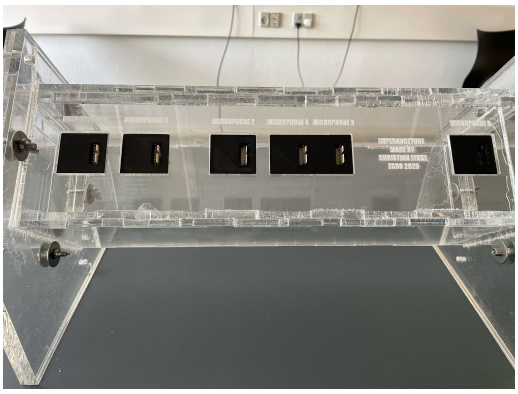
A clear view into the sample holder is achieved, and it is rather easy to determine if the sample is placed correctly. The coupling between the sample holder and microphone tube serves as a visual guide for placing the sample, as the front edge should follow the coupling on all four sides. To actually lift the lid of the sample holder, two bolts have been placed in the acrylic as an impromptu handle.

Adjusting the terminating end is deceptively easy, as it is simply placed behind the sample, leaving little room for error.

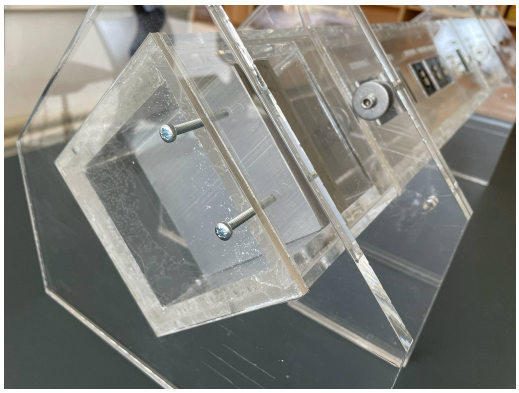
In figure 5.8 several images of the final tube can be seen:



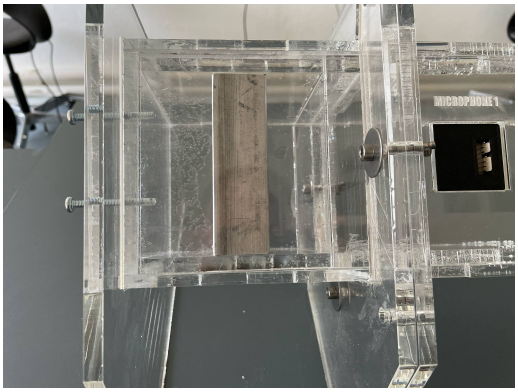
(a) Complete impedance tube assembled - without MCU, cables, and speaker.



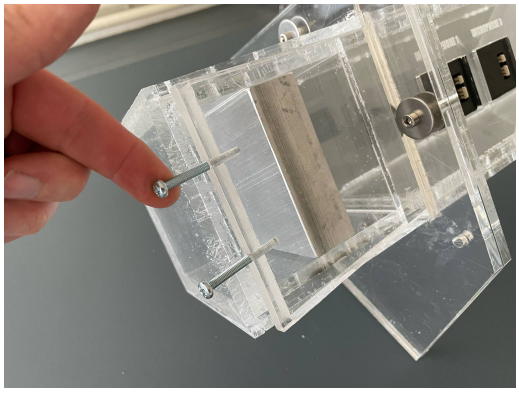
(b) Closer look at the microphone tube.



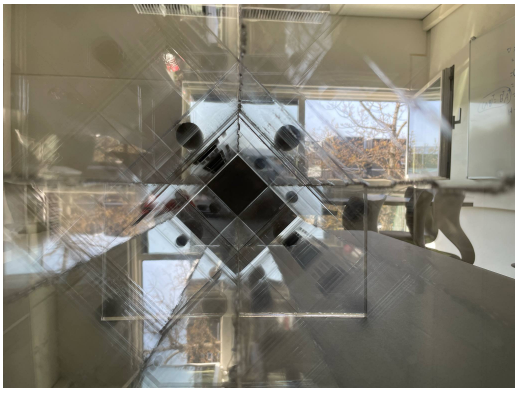
(c) Closer look at the sample holder.



(d) Closeup of the sample holder with terminating end placed within.



(e) Sample holder lid being lifted by impromptu mounted bolts.



(f) Closeup inner tube of the impedancetube, where it can be seen to be smooth.

Figure 5.8: Various images of the final impedance tube.

Test Summary

As mentioned in the test results, the physical tube complies with all demands other than 3 measurements. However, these measurements are still within tolerances posed by the DS ISO 10534:2023 standard. In table 5.3 all test results can be read, with their related deviance, and whether or not the given criteria has been met. Despite the 3 measurements being out of tolerance, the tube as a whole has been accepted, and will therefore be used for all subsequent tests wherein it can be used. It is very important to notice that the measurements from speaker to microphone placements are 10mm shorter in this table, than in reality. This is due to the speaker membrane being placed 10mm within the speaker cabinet, as mentioned the beginning of section 5.1.8.

Test Criteria	Expected	Measured	Deviation (%) & (mm)	Accepted/ Denied
Length Sample Holder	100	100.19	0.19% & 0.19	Accepted
Width Sample Holder (Inner)	100	100.09	0.09% & 0.09	Accepted
Width Sample Holder (Outer)	128	127.73	0.21% & 0.27	Denied
Height Sample Holder (Inner)	100	99.78	0.22% & 0.22	Denied
Height Sample Holder (Outer)	128	127.78	0.17% & 0.22	Accepted
Cross Measure 1 Sample Holder	141.42	141.28	0.10% & 0.14	Accepted
Cross Measure 2 Sample Holder	141.42	141.70	0.20% & 0.28	Accepted
Length Microphone Tube	450	449.8	0.04% & 0.2	Accepted
Width Microphone Tube (Inner)	100	99.98	0.02% & 0.02	Accepted
Width Microphone Tube (Outer)	128	127.81	0.15% & 0.19	Accepted
Height Microphone Tube (Inner)	100	100.09	0.09% & 0.09	Accepted
Height Microphone Tube (Outer)	128	128.39	0.31% & 0.39	Denied
Cross Measure 1 Microphone Tube	141.42	141.21	0.15% & 0.21	Accepted
Cross Measure 2 Microphone Tube	141.42	141.15	0.19% & 0.27	Accepted
Length Blank Tube	374.39	374.1	0.08% & 0.29	Accepted
Width Blank Tube (Inner)	100	99.98	0.02% & 0.02	Accepted
Width Blank Tube (Outer)	128	127.95	0.04% & 0.05	Accepted
Height Blank Tube (Inner)	100	100.01	0.01% & 0.01	Accepted
Height Blank Tube (Outer)	128	128.08	0.06% & 0.08	Accepted
Cross Measure 1 Blank Tube	141.42	141.20	0.16% & 0.22	Accepted
Cross Measure 2 Blank Tube	141.42	141.52	0.07% & 0.10	Accepted
Microphone 1 Placement	753.68	753.6	0.1% & 0.08	Accepted
Microphone 2 Placement	626.39	626.3	0.01% & 0.09	Accepted
Microphone 3 Placement	701.68	701.5	0.03% & 0.18	Accepted
Microphone 4 Placement	574.39	574.2	0.03% & 0.19	Accepted
Microphone 5 Placement	541.55	541.3	0.05% & 0.25	Accepted
Microphone 6 Placement	414.26	414.2	0.01% & 0.06	Accepted
Sealable Microphone Sockets	-	-	-	Accepted
Sealable Sample Holder	-	-	-	Accepted
Airtight Connection With Type 4002 speaker	-	-	-	Accepted
Clear View of the Sample	-	-	-	Accepted
Easy Adjustment of the Terminating End	-	-	-	Accepted

Table 5.3: Result overview for physical compliance of the tube. All measurements are in mm.

5.2 Creating Sound

As mentioned in the technical analysis, the signals that will be used to determine acoustic criteria are MLS, sine sweeps, and pure sinusoids. The signals created will be outputted by a 3.5 mm jack, making the system easily connected to speakers/amplifiers and thus creating sound.

In general, this section requires the use of:

- Teensy 4.1
- Teensy PT8211 Audio Kit
- Teensy Audio Shield
- Speaker with jackstick connectivity.

Moreover, each signal type's functional frequency band is limited from 0 to 1212.69Hz, as the tube's construction does not permit measurements at higher frequencies, derived in equation 5.2.

5.2.1 Inputting A Signal

Before any signals are made, it has to be determined how the signal should be converted into a signal that the speaker can play. Achieving this is done by the I2S protocol. To actually make this element, known test signals are used and implemented. The module is designated to convert bitstreams into a signal, which can be played through a speaker.

Specification

- I2S compatability
- Appropriate speaker output
- Control 4 signals at once
- Create a signal true to the original sample

Design

The connection between creating a signal and playing it as audio is bridged by the comprehensive audio capabilities of the Teensy. By importing the "Audio.h" header, several functions become available. Therefore, this module is not designed per se, but rather utilizes the possibilities found in the "Audio.h" header file.

Implementation

As 4 signals is desired to use for testing, it would be beneficial to encompass all at once. Teensy has a built-in mixer when using "Audio.h", which enables exactly that. The "AudioMixer4" is set to connect via I2S to the audio shield. Then, each following signal type is given a channel in the mixer. For the MLS signal, "AudioPlayQueue" is used, as it accepts binary arrays as input. To create a sine wave, "AudioSynthWaveform" is used, as it creates a sine wave matching a given frequency and phase. White noise is made by the "AudioSynthNoiseWhite" function. The sine sweep is made with "AudioSynthToneSweep".

Each signals gain can be adjusted either in their separate functions, or at the mixer. To make the functionality more intuitive, gain will only be adjusted at the mixer.

As all signals can now be played through built-in functions, further utilization should be rather easy.

Test

The purpose of the tests is to ensure that a signal can be played by itself on each channel. Tests are conducted using the expected audio outputs. Furthermore, the purpose of the test is to find a suitable sound level for all further testing of each specific signal type.

Test Setup To perform the test, the following is needed:

- Teensy 4.1
- USB-A to micro-USB cable
- SD card for CSV data obtainment
- PC for code upload and data handling
- Speaker for listening test
- AUX-cord
- MATLAB script to check MLS for correctness

Actual Testing No remarks to actual testing, besides that gains are set very low to play audio at a pleasant level.

Test Results The test was conducted without any errors. The signal processing is compatible with I2S and the audio shield, making it applicable for use. Furthermore, the mixer is capable of handling 4 separate signals at once, controlling them individually while also controlling left/right stereo. By ear, all signals seem to be played correctly.

Summary

All tested signals can be input to the Teensy and played on a speaker using the audio shield and an AUX cord. Therefore, it is now possible to analyze each signal type in depth. To do further analysis, the following function has been developed to save the signals in .CSV format, sampling at 44.1kHz in 128 bit block sizes:

```
1  const int SAMPLE_RATE = 44100;                          // Default sample rate
2  const int BLOCK_SIZE = 128;                             // Default block size
       for the audio.h library
3  const int BLOCKS_NEEDED = SAMPLE_RATE / BLOCK_SIZE;  // 1 second of samples
4
5  void recordAudioToFile() {
6    int blocksRecorded = 0;
7    while (blocksRecorded < BLOCKS_NEEDED) {               // Ensures that
       enough measurements are made to fill 1 second of analysis
8      if (recordQueue.available() > 0) {                   // Check if any
       audio is even available
9        int16_t* buffer = recordQueue.readBuffer();        // Reads from the
       queue
10       for (int i = 0; i < BLOCK_SIZE; i++) {             // Loop through
       128 times to comply with Audio.h standards
11         float sample = (float)buffer[i] / 32768.0f;      // Normalize
       sample to float range (-1, 1)
12         logFile.print(sample, 6);                        // Saves sample
       with 6 decimalpoints
13         logFile.print(i < BLOCK_SIZE - 1 ? "," : "\n");  // Separate values
        with commas to comply with .csv format
14       }
15       recordQueue.freeBuffer();  // Free buffer before next block is
       started
16       blocksRecorded++;
17     }
18   }
```

```
}
```

Full code implementation can be found on GitHub here or in the attached files as "XXXX".

5.2.2 MLS

The MLS signal is the main signal type which will be used to determine acoustic characteristics, as its broadband characteristics reveals all there is to know about a samples capabilities.

Specification

- Capable of creating 2 to 32-bit signals
- Adjustable bit length based on need
- Must execute in real-time
- Must be compatible with I2S

Design

The MLS signal is created by a simple C script, which can be accessed on GitHub here or in the attached files. As the MLS generator is implemented directly in C programming, the physical characteristics of an LFSR are not considered. The MLS generator functions as described in section 3.4.2, taking a non-zero starting input (in this case, all 1s), and XOR'ing at taps matching the number of bits. The program has been made to support bit lengths from 2 to 32 bits, making it possible to adjust the MLS based on the current needs.

Implementation

The MLS generator is implemented as a standalone function in the Arduino IDE. The function creates a seed consisting of all 1s to start the MLS generation. This makes it obvious to determine when a "new round" of the MLS has begun. Then, a switch case determines the placement of feedback taps, ensuring an MLS is created. As the MLS is generated, it is outputted to the 3.5mm jack, which is described in detail in section 5.2.1. Apart from the actual MLS generation, timers are added as well, to measure performance metrics. Pseudocode of the entire function can be seen here:

```
// Feedback tap map for various LFSR lengths (primitive polynomials)
uint32_t feedbackTaps(uint8_t bits) {
  switch (bits) {
    case 2: return (1 << 1) | (1 << 0);
    case 3: return (1 << 2) | (1 << 0);
    // ...
    case 32: return (1 << 31) | (1 << 21) | (1 << 1) | (1 << 0);
  }
}

void generateMLS() {
mask = (1UL << LFSRBits) - 1;  // 1UL creates a Unsigned Long consisting of
    pure 1's. When bitshifted, this creates the mask/seed which starts the
    MLS as N bits of 1's
LFSR = mask;                   // Initial state must be non-zero
uint32_t taps = feedbackTaps(LFSRBits);
uint32_t MLSLength = (1UL << LFSRBits) - 1;

  for (uint32_t i = 0; i < MLSLength; i++) {
    bool feedback = __builtin_parity(LFSR & taps);  // Parity of taps
```

```
    Serial.print(feedback ? 1 : 0); // Output the feedback bit, NOT the LFSR MSB
    LFSR <<= 1;  // Shift left
    if (feedback) {
        LFSR |= 1;  // Insert feedback at LSB
    }
    LFSR &= mask;  // Mask to keep LFSRBits width
}
```

To play the generated MLS signal, the following function is introduced:

```
void playMLSBit(bool bit, int samplesPerBit = 1, int amplitude = 28000) {
    // Bool is the actual input parameter of the function (0 or 1), while samplesPerBit and amplitude are static. Amplitude is set at 28000 to not risk clipping the signal on a speaker
  static int16_t buffer[128];
  static int bufferIndex = 0;
  int16_t value = bit ? amplitude : -amplitude;  // Ternary operator deciding to set the amplitude at + or - given amplitude based on boolean value
  for (int i = 0; i < samplesPerBit; i++) {
    buffer[bufferIndex++] = value;
    if (bufferIndex == 128) {                                    // Buffer set to 128 bits, conforming to I2S standards
      memcpy(MLSSignal.getBuffer(), buffer, sizeof(buffer));  // Copies buffer data to the AudioPlayQueue, enabling the Teensy to play the MLS
      MLSSignal.playBuffer();                                    // Proprietary play function for the AudioPlayQueue
      bufferIndex = 0;                                           // Resets the bufferindex, so a new sequence can be initialized
    }
  }
}
```

Full code implementation can be found on GitHub here or in the attached files as "XXXX".

Test

The test aims to evaluate the correctness and determine the execution time of MLS generation on the Teensy as a standalone program.

Test Setup To perform the test, the following is needed:

- Teensy 4.1
- Teensy Audio Shield
- USB-A to micro-USB cable
- SD card for CSV data obtainment
- PC for code upload and data handling
- Speaker for listening test
- AUX-cord
- MATLAB script to check MLS for correctness

Actual Testing The first tests revealed that the generated MLSs did not have the desired characteristics. Several issues arose, making the generated MLS riddled with faults. The main cause of these faults was that each LFSR's outputted bit was the MSB, instead of the feedback bit generated by the taps. Therefore, to the naked eye, each MLS seemed correct, but was faulty when played. This led to the creation of a MATLAB script specifically for testing the MLS against the laws set in 3.4.2 on page 44, which can

be accessed on GitHub here or in the attached files as "MLSChecker.m". The checker has been tested with known faulty MLSs. The known faulty MLSs were generated by both the integrated MLS function in MATLAB and by the Teensy. Common in all faulty codes was the change of a single bit chosen at random, and in each sequence, the checker was capable of discerning right from wrong. However, to check the sequences, it has been necessary to log them in .txt files, which can be found here or seen in the attached documents. When the code had been refactored, some sequences continued to be faulty. The error was found to be wrongful taps being set, leading to incomplete MLSs. However, after rewriting the program and checking the generated sequences in MATLAB again and again, the MLS generator functions as expected.

Test Results In figure 5.9 plots of the MLS sequence, autocorrelation, FFT magnitude, and spectral density of a 16-bit MLS can be seen, generated by the Teensy, using the same sequence with a single randomly chosen bit flipped, the MLSChecker script successfully notifies that the MLS is not correct, even though no difference can be seen in any of the plots. Printouts from the MLSChecker at 16 bits can be seen below the figures. Comparing the Teensy MLS with the MLS generated by MATLAB in figure 5.11, it is evident that the correct MLS generated on the Teensy is applicable for further use. In appendix "MLS Test Outputs" starting page 128, similar results can be seen for 4-bit to 30-bit. The 31 and 32-bit analyses have been left out, as system requirements to complete the analysis at the same precision level exceed the capabilities of my laptop. The only evaluation made for these is that they sound like white noise, which, as shown by the faulty sequences, isn't a good metric.

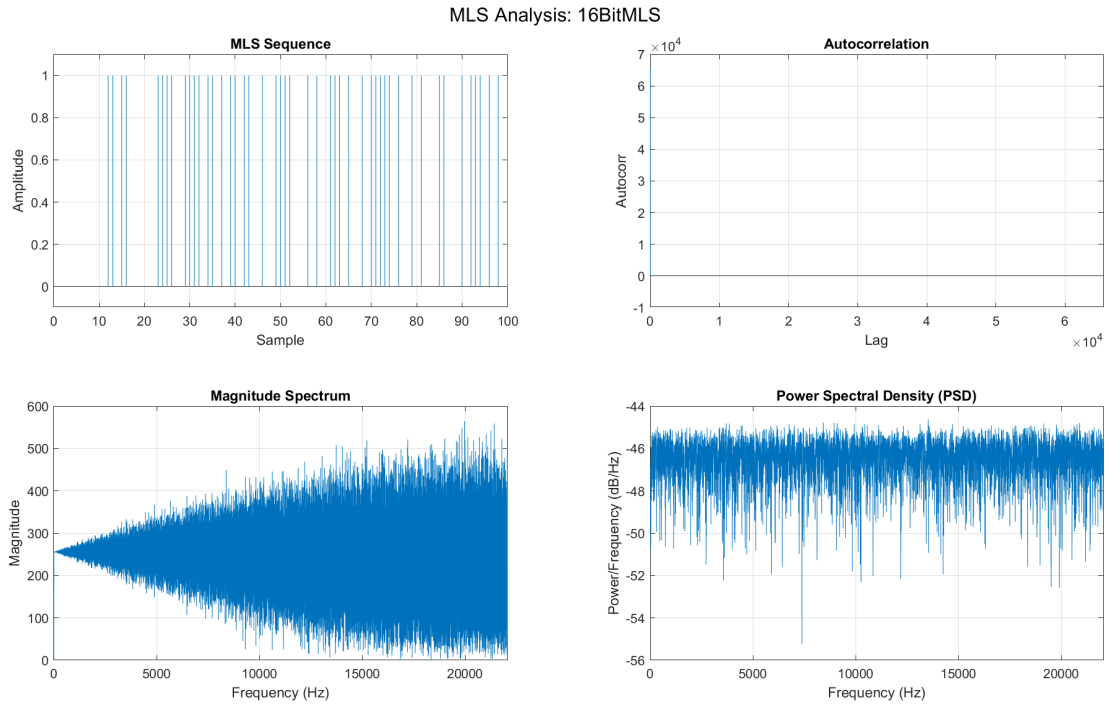


Figure 5.9: Plots related to the 16-bit MLS sequence generated on the Teensy.

Sequence length: 65535 samples
Detected register size: 16 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:

Length : 65535 samples
Register size (n) : 16 bits
Detected cyclic shift : 0
Autocorrelation peak : 65535.000000 (ideal = 65535)
Max sidelobe deviation: 0.000000

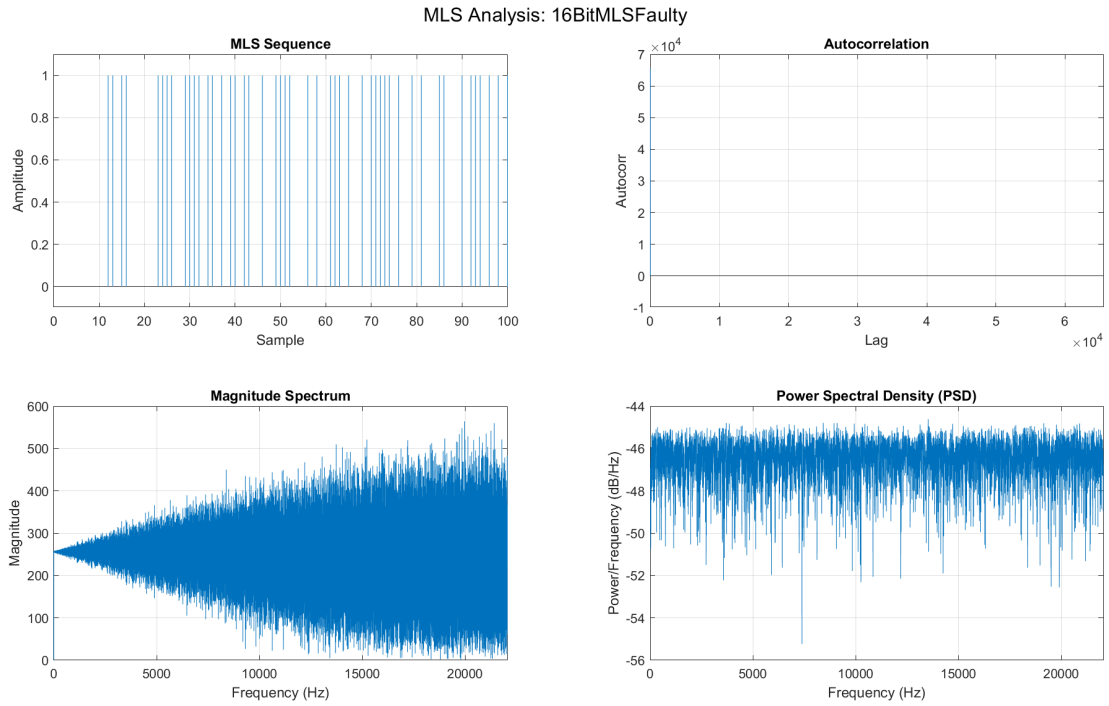


Figure 5.10: Plots related to the faulty 16-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 65535 samples
Detected register size: 16 bits
✖Sequence FAILED MLS check.
Summary:
Length : 65535 samples
Register size (n) : 16 bits
Detected cyclic shift : 0
Autocorrelation peak : 65535.000000 (ideal = 65535)
Max sidelobe deviation: 4.000000

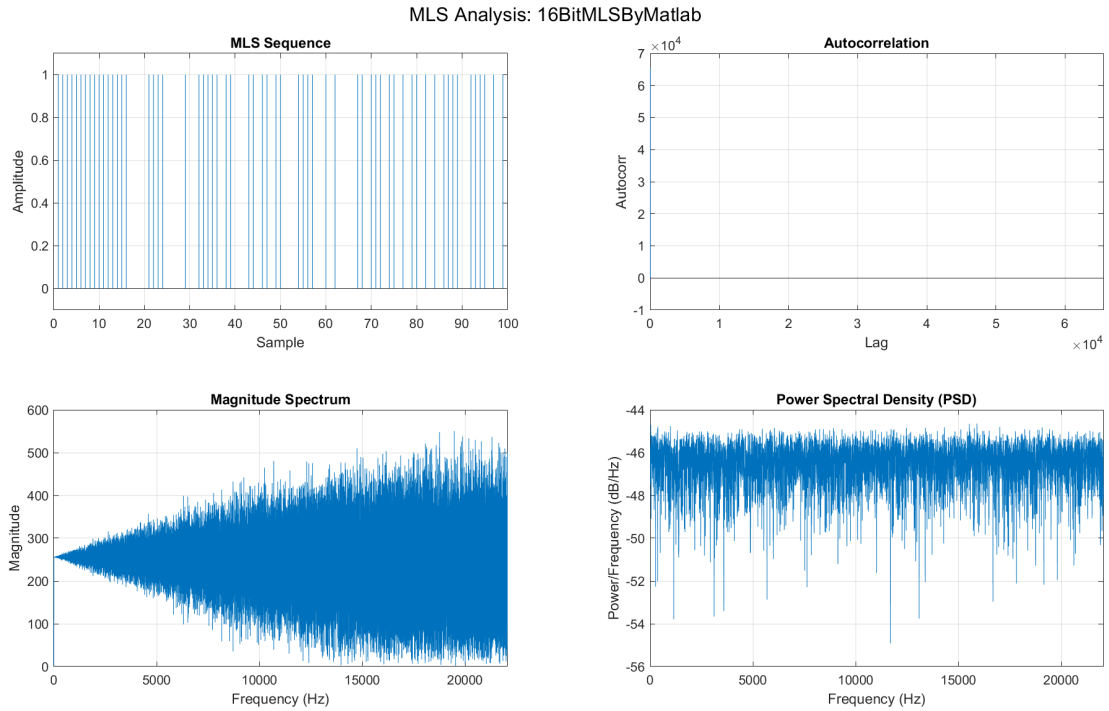


Figure 5.11: Plots related to the 16-bit MLS sequence generated by MATLAB.

In table 5.4, average execution times for the generation and playback of each MLS are seen at a sample rate of 44.1 kHz. Each average is obtained across 5 samples, until 24 bits, as it would take 270 hours[1] to get 10 execution times at 32 bits. For 25-32 bits, a single run is timed.

No. of bits	2 bits	3 bits	4 bits	5 bits	6 bits	7 bits	8 bits
Time in S	$1.00 * 10^{-6}$	$1.60 * 10^{-6}$	$1.60 * 10^{-6}$	$2.60 * 10^{-6}$	$4.60 * 10^{-6}$	$8.40 * 10^{-6}$	$14.6 * 10^{-6}$
9 bits	10 bits	11 bits	12 bits	13 bits	14 bits	15 bits	16 bits
$27.60 * 10^{-6}$	$53.00 * 10^{-6}$	$106.4 * 10^{-6}$	$208.2 * 10^{-6}$	$418.4 * 10^{-6}$	$141.15 * 10^{-3}$	$512.34 * 10^{-3}$	$1.26 * 10^{0}$
17 bits	18 bits	19 bits	20 bits	21 bits	22 bits	23 bits	24 bits
$2.74 * 10^{0}$	$5.71 * 10^{0}$	$11.66 * 10^{0}$	$23.55 * 10^{0}$	$47.3 * 10^{0}$	$94.89 * 10^{0}$	$189.99 * 10^{-6}$	$380.20 * 10^{0}$
25 bits	26 bits	27 bits	28 bits	29 bits	30 bits	31 bits	32 bits
$760.63 * 10^{0}$	$1.52 * 10^{3}$	$3.04 * 10^{3}$	$6.09 * 10^{3}$	$12.17 * 10^{3}$	$24.34 * 10^{3}$	$48.69 * 10^{3}$	$97.39 * 10^{3}$

Table 5.4: Average execution times of MLS generation and playback on the Teensy 4.1 in seconds at a sample rate of 44.1kHz. For bits 25-32, only 1 run has been timed due to their long execution times. A complete table with raw values is available in appendix I.1 on page 128.

Summary

All in all, the MLS generator has been difficult to get functional. However, the current version found here is fully functional and creates correct MLSs. The generator is capable of generating verified MLSs of 2-30 bits, and is anticipated to create correct

[1]Estimated based on 1 32-bit run measured at almost 27 hours.

31 and 32-bit MLSs. Furthermore, the MLSchecker is capable of detecting a single bit being flipped, creating confidence that the current MLS generation is correct.

5.2.3 Pure Sinusoids

To create synonymous measurement capabilities to the Type 4002 impedance tube, pure sinusoids should be generatable as well.

Specification

- Create a pure sinusoid at frequencies from 20-1220 Hz
- Be able to phase shift the signal

Design

As pure sinusoids can be created directly using the "Audio.h" library, and specifically the AudioSynthWaveForm function, the only design choices made for this module is a for loop iterating from 20-1220 Hz, along with a for loop iterating 90-degree phase shifts at 1 kHz. The sine generator will be connected to channel 1 in the mixer.

Implementation

The implementation is best shown through code snippets:

```
AudioSynthWaveform sineWave;                          // Utilizes pre-made sine wave generator
AudioConnection patchCord2(sineWave, 0, mixer, 1);    // Sends the pure sine to channel 1 in the mixer

void recordSine() {
  Serial.println("Recording sine wave");
  sineWave.begin(WAVEFORM_SINE);          // Specifies a sine wave. Could also be sawtooth, square, triangle etc
  sineWave.amplitude(0.3);                // Sets the gain/amplitude of the sinusoid
  for (int f = 20; f <= 1220; f += 50) {  // Increments frequency by 50 Hz
    sineWave.frequency(f);                // Sets the current frequency
    char filename[20];
    snprintf(filename, sizeof(filename), "sine%d.csv", f);
    logFile = SD.open(filename, FILE_WRITE);  // Creates file on the SD card
    recordAudioToFile();                      // Saves the current audio to SD
    logFile.close();                          // Closes the file on SD
  }
  sineWave.amplitude(0);  // Turns off the sinusoid
  Serial.println("Sine done");
}

void recordPhaseShift() {
  Serial.println("Recording phase shift at 1kHz");
  sineWave.amplitude(0.3);                            // Sets the gain/amplitude of the sinusoid
  sineWave.frequency(1000);                           // Sets the frequency
  for (int phase = 0; phase <= 360; phase += 90) {  // Increments phase shift by 90 degrees
    sineWave.phase(0);                                // Initializes phase at 0 degrees
    char filename[20];
```

```
    snprintf(filename, sizeof(filename), "sineShifted%d.csv", phase);
    logFile = SD.open(filename, FILE_WRITE);  // Creates file on the SD
    card
    recordAudioToFile();                      // Saves the current audio to
     SD
    logFile.close();                          // Closes the file on SD
  }
  sineWave.amplitude(0);  // Turns off the sinusoid
  Serial.println("Phaseshift done");
}
```

Full code implementation can be found on GitHub here or in the attached files as "XXXX".

Test

The purpose of the test is to ensure that the Teensy is capable of creating pure sinusoids, and is capable of phase-shifting them. The test is done by running the code, sampling the sinusoid on an SD card, and letting the MATLAB script "teensySignalAnalysis.m" plot results.

Test Setup To perform the test, the following is needed:

- Teensy 4.1
- Teensy Audio Shield
- USB-A to micro-USB cable
- SD card for CSV data obtainment
- PC for code upload and data handling
- Speaker for listening test
- AUX-cord

Actual Testing During testing, aliasing was thought to occur, due to a mismatch between the .CSV formatting on the Teensy, and .CSV reading in MATLAB. This resulted in quite a few hours spent trying to fix an aliasing problem which didn't exist.

Test Results As can be seen from figure 5.12 and 5.13, the Teensy is perfectly capable of generating a pure sinusoid at any given frequency. A phase shifted signal at 1 kHz can be seen in figure 5.14.

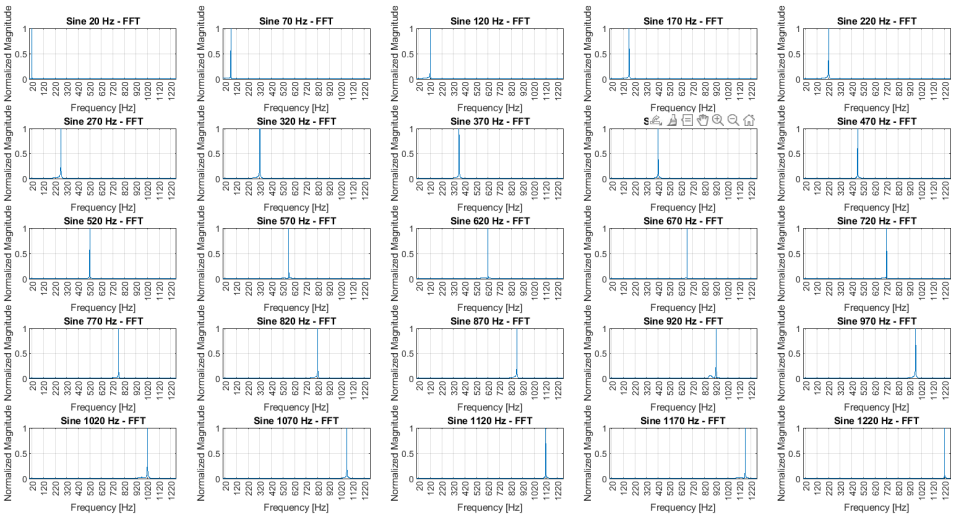


Figure 5.12: Collected FFTs of pure sinusoids generated by the teensy, incrementing 50 Hz from 20Hz to 1220Hz.

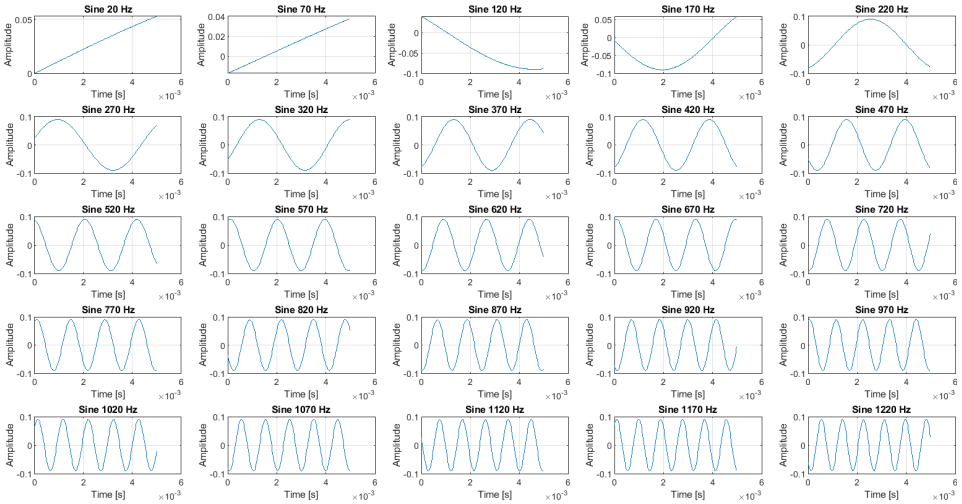
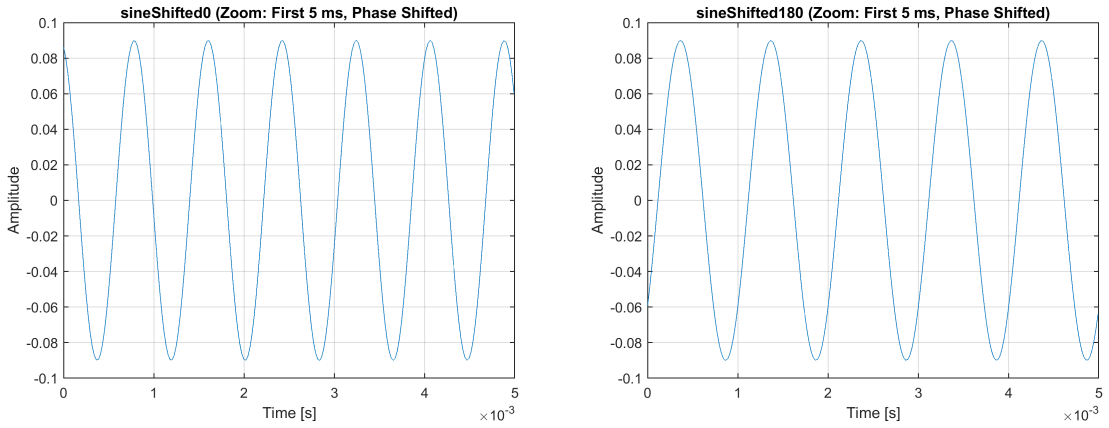


Figure 5.13: Collected time domain plots of pure sinusoids generated by the teensy, incrementing 50 Hz from 20Hz to 1220Hz.



(a) 0 degree phase shifted sinusoid at 1 kHz.

(b) 180 degree phase shifted sinusoid at 1 kHz.

Figure 5.14: Comparison of phase shifted sinusoids at 1 kHz.

Summary

From the pure sinusoid testing, it is confirmed that the Teensy is capable of creating puretones and is capable of phaseshifting the signal at will.

5.2.4 White Noise

If a truly random signal is desired, white noise is the way to go. White noise contains most of the same characteristics of an MLS, however without the deterministic knowledge, making the signal truly random.

Specification

- Create white noise compliant with general consensus

Design

A white noise generator can be found in the "Audio.h" library, and implemented through the AudioSynthNoiseWhite function. As white noise is specified strictly, no effort will be made to modify the white noise generator.

Implementation

The white noise generator is implemented as shown in the following code snippet. Do note that is pink noise is desired instead of white noise, it can easily be created by interchanging "AudioSynthNoiseWhite" with "AudioSynthNoisePink". The remaining code is identical.

```
// When white noise is generated
AudioSynthNoiseWhite whiteNoise;    // Utilizes pre-made white noise
    generator
AudioConnection patchCord3(whiteNoise, 0, mixer, 2);    // Sends the white
    noise to channel 2 in the mixer

void recordWhiteNoise() {
  Serial.println("Recording white noise");
  whiteNoise.amplitude(0.3);    // Sets the amplitude for white noise
    signal
  logFile = SD.open("white.csv", FILE_WRITE);   // Creates a file on the SD
     card
  recordAudioToFile();  // Saves the current audio to SD
  logFile.close();  // Closes the file on SD
  whiteNoise.amplitude(0);  // Turns off the white noise
  Serial.println("White noise done.");
}
```

Full code implementation can be found on GitHub here or in the attached files as "XXXX".

Test

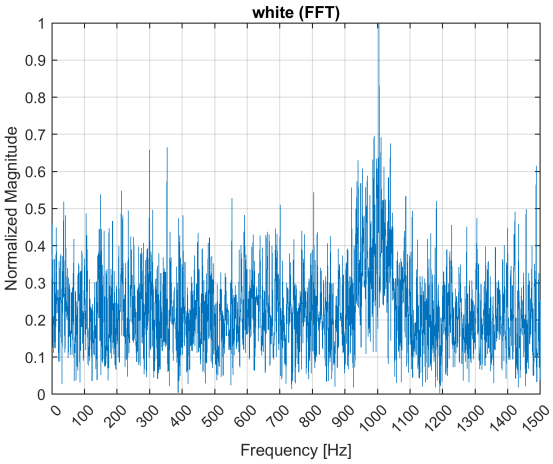
The test is supposed to show that the white noise generator creates equal random noise, with a tendency towards higher frequencies.

Test Setup To perform the test, the following is needed:

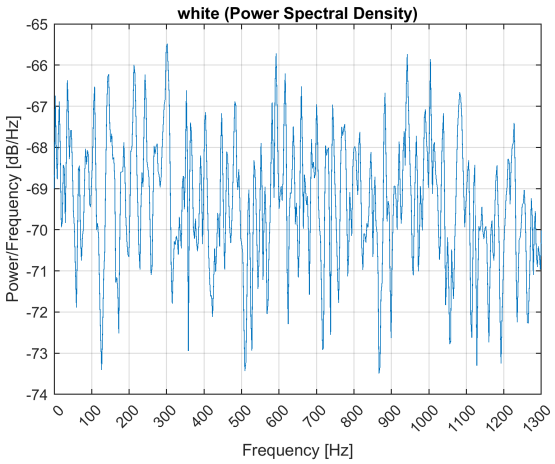
- Teensy 4.1
- Teensy Audio Shield
- USB-A to micro-USB cable
- SD card for CSV data obtainment
- PC for code upload and data handling
- Speaker for listening test
- AUX-cord

Actual Testing When listening to the white noise generated, it sound exactly like the signal produced by the MLS. Otherwise, no noteworthy comments.

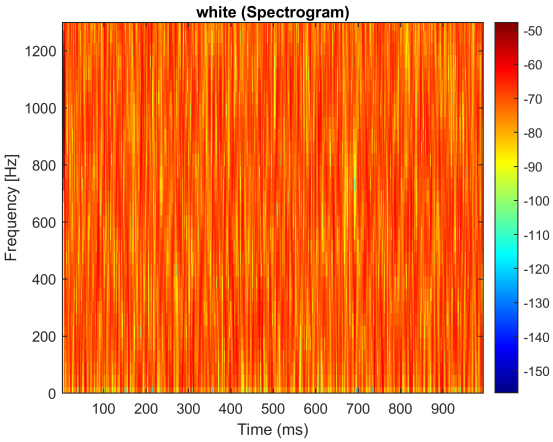
Test Results The white noise generation has shown to be somewhat compliant with general consensus of what white noise should be. However, a strong peak is visible at 1kHz in figure 5.15a, suggesting that the white noise generator does not create truly random noise.



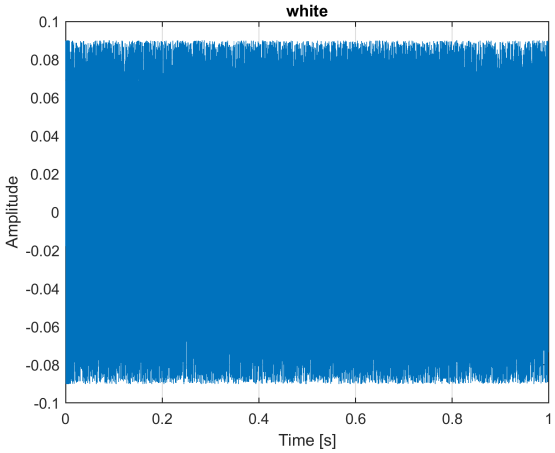
(a) FFT of the white noise signal created by the Teensy.



(b) Spectral density of white noise created by the Teensy.



(c) Spectrogram of white noise created by the Teensy.



(d) White noise created by the Teensy in the time domain.

Figure 5.15: Plots of sweep characteristics.

Summary

While the white noise generator sounds like white noise, the signal does not fit exactly within the general consensus of what white noise should be. This is due to the peak at 1 kHz, and the white noise generator will therefore only be used "for fun" in future tests.

5.2.5 Sine Sweep

The sine sweep signal type is added to the available signal types to allow testing based on personal preferences. While the sweep contains no more data than the MLS or white noise signal, it can for some uses be preferable, such as analyzing a very specific frequency band.

Specification

- Must be capable of sweeping given frequencies
- Must be capable of sweeping within 20Hz to 1220Hz
- Must be able to perform sweep in a given timeframe
- Must create a linear sine sweep

Design

The sine sweep is designed around the integrated AudioSynthToneSweep function from the "Audio.h" library. The function takes gain, start frequency, end frequency, and duration as inputs, and will be connected to channel 3 on the mixer.

Implementation

As for previous integrated functions, the implementation is best shown through a code snippet:

```
AudioSynthToneSweep sineSweep;                          // Utilizes pre-made
    sine sweep generator
AudioConnection patchCord4(sineSweep, 0, mixer, 3);  // Sends the sine
    sweep to channel 3 in the mixer

void recordSineSweep() {
  Serial.println("Recording sine sweep...");
  sineSweep.play(1, 20, 1220, 1);  // Initializes sine sweep from 20Hz to
    1220Hz over 1 second
  logFile = SD.open("sweep.csv", FILE_WRITE);  // Creates a file on the SD
    card
  recordAudioToFile();                          // Saves the current audio
    to SD
  logFile.close();                              // Closes the file on SD
  Serial.println("Sine sweep done.");
}
```

Full code implementation can be found on GitHub here or in the attached files as "XXXX".

Test

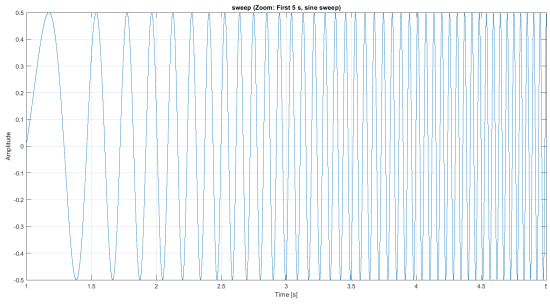
Testing the sine sweep is done to ensure that a sine sweep behaves exactly as instructed. The test will verify frequencies, duration, and linearity.

Test Setup To perform the test, the following is needed:

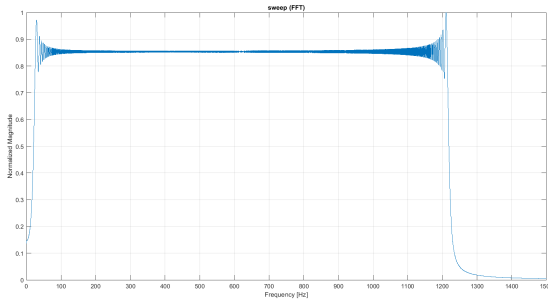
- Teensy 4.1
- Teensy Audio Shield
- USB-A to micro-USB cable
- SD card for CSV data obtainment
- PC for code upload and data handling
- Speaker for listening test
- AUX-cord

Actual Testing No noteworthy comments.

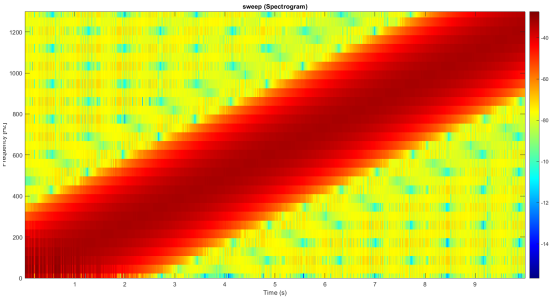
Test Results In figure 5.16, characteristics of the sine sweep can be seen. It has been shown that the Teensy is capable of creating a linear sine sweep, and that it with few parameters changes can be adapted to different frequency bands and durations.



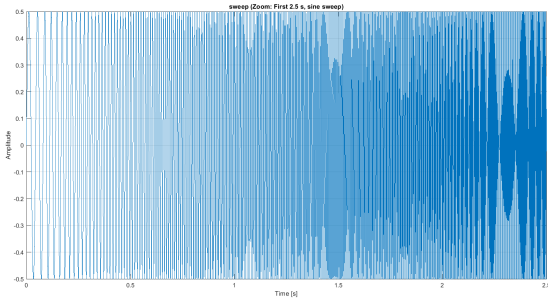
(a) Time domain of sine sweep from 1Hz to 50 Hz over 10 seconds (first 5 shown), created by Teensy..



(b) FFT of sine sweep from 20Hz to 1220Hz over 10 seconds, created by the Teensy.



(c) Spectrogram of sine sweep from 20Hz to 1220Hz over 10 seconds showcasing the linear properties of the sweep, created by the Teensy.



(d) Zoomed time domain of sine sweep from 20Hz to 1220Hz over 10 seconds, created by the Teensy.

Figure 5.16: Plots of sweep characteristics.

Summary

A sine sweep can be made using the Teensy, and it is quite easy to design and utilize.

5.3 Measuring Sound

To facilitate any measurements in the impedance tube, the system must be capable of measuring sound efficiently with great precision. As mentioned in section 5.1.6, the PUI Audio DMM-4026-B-I2S-R microphones have been chosen to facilitate this. The sound measuring design as a whole will be divided into 3 sections: measuring with 1, 2, and 6 microphones. To keep a focus, a specification for the section as a whole is made. The measuring system should be able to:

- Measure from 20-1220 Hz
- Measure at two microphones at a time
- Measure broadband signals
- Measure impulse responses
- Measure FFT's
- Measure at all 6 microphones

The demands can be extrapolated onto each subsection in this design phase, as each setup of microphones must be capable of fulfilling these.

5.3.1 Measuring With 1 Microphone

To approach the demands systematically, measuring FFTs and impulse responses at 1 microphone is refined first. Improving measurements at 1 microphone first will yield knowledge and insight into how the system can be expanded to multiple microphones.

Design

Measuring an FFT on the teensy is rather easy, as it has several DSP functions built in. One of those is 256 and 1024-bit FFTs, which even include the most common window functions, except for a rectangular window. This is troublesome, as MLS signals are periodic and no spectral leakage is made by the signal itself. Therefore, applying any type of window would distort the spectral properties of the MLS signal, resulting in skewed/wrongful results, while introducing artifacts to the FFT, which aren't present in the original signal. Therefore, to mitigate these factors, the "analyze_fft1024.h" library is modified along with the "analyze_fft1024.cpp" file to include a rectangular window. The additions made in the .cpp file are:

```
1 const int16_t AudioWindowRectangular1024[1024] = { //fills all 1024 entries
      with 32767, equalling 1 in q-format
2 32767, ... , 32767};
```

And in the .h file:

```
1 extern "C" {
2 extern const int16_t AudioWindowOther1024[];
3 extern const int16_t AudioWindowRectangular1024[];
4 }
```

With these additions, a rectangular window can now be applied to the FFT function. A Hanning window is chosen to gain the sharpest possible frequency analysis for pure-tone sinusoid analysis, even though a flattop window has also been considered to increase amplitude accuracy. For white noise, the Hanning window type has been chosen on the basis of averaging measurements, increasing SNR as discussed in 3.4.2 on page 46. When analyzing linear chirps, a Hanning window has been chosen as well, to get decent sidelobe attenuation with sufficient frequency precision. By utilizing these windows, proper FFT measurements should be possible.

Measuring the impulse response requires that the expected signal is measured simultaneously with the actual played signal. This will be achieved by reading the signal directly at the mixer introduced in 5.2, and at the microphones' I2S channel. Besides measuring the signals, they are also saved in .CSV files, for future data interpretation.

Implementation

Implementing FFT measurements are done simply by utilizing the libraries mentioned, as can be seen in the following code snippet:

```
AudioInputI2S            i2sMic;
AudioAnalyzeFFT1024      fft1024;
AudioConnection          patchCord1(i2sMic, 0, fft1024, 0);
const float binWidth = 44100.0 / 1024.0;  // ~ 43.07 Hz bandwidth of bins
void setup() {
  Serial.begin(115200);
  AudioMemory(12);
  fft1024.windowFunction(AudioWindowRectangular1024);
}
void loop(){
if (fft1024.available()) {
    for (int i = 0; i < 40; i++) {
      float startFreq = i * binWidth;
      float endFreq = (i + 1) * binWidth;
      float val = fft1024.read(i);
        // Serial prints to make interpretation easy
      Serial.print(startFreq, 1);
      Serial.print("-");
      Serial.print(endFreq, 1);
      Serial.print(": ");
      Serial.print(val, 6);
      Serial.print("  ");
    }
    Serial.println();
  }
}
```

Measuring the impulse response comes with a large set of challenges, the first being to ensure both the generated signal and measured signal are timed correctly. The first approach made was implementing a function that would record both signals to separate .CSV files simultaneously. However, writing to separate files on an SD card at 44.1 kHz comes with certain challenges. Therefore, the function was refactored to save both input streams to a single .CSV file:

```
void recordBothToFileSingleFile(File& file, uint32_t totalSamples) {
  uint32_t samplesRecorded = 0;                                        //
    Checker for when to stop
  while (samplesRecorded < totalSamples) {
    if (recordQueue.available() && mic1Queue.available()) {  // Checks for
    data on both signal lines, ensuring that data is present on both lines
    simultaneously
      int16_t* bufMixer = recordQueue.readBuffer();
      int16_t* bufMic = mic1Queue.readBuffer();
      for (int i = 0; i < blockSize; i++) {  // Divides signals into blocks
     to ensure no overflow is happening
        float sMixer = (float)bufMixer[i] / 32768.0f;
        float sMic = (float)bufMic[i] / 32768.0f;
        file.print(sMixer, 6);  // Saves mixer signal to SD card
        file.print(",");        // Saves to SD card
        file.print(sMic, 6);    // Saves mic signal to SD card
```

```
13          file.print("\n");       // Saves to SD card
14        }
15        recordQueue.freeBuffer();      // Frees buffer
16        mic1Queue.freeBuffer();        // Frees buffer
17        samplesRecorded += blockSize;  // Keeps track of how many samples has
      been saved
18      }
19    }
20  }
```

By doing so, data has successfully been logged, and can now be interpreted by software such as MATLAB. Full code implementation can be found on GitHub here or in the attached files as "XXXX".

Test

The purpose of the test is to verify that the Teensy is capable of creating an FFT and measuring an impulse response using 1 microphone

Test Setup To perform the test, the following is needed:

- Teensy 4.1
- Teensy Audio Shield
- USB-A to micro-USB cable
- SD card for .CSV data obtainment
- PC for code upload and data handling
- Speaker for listening test
- AUX-cord
- Tone generator
- PUI Audio DMM-4026-B-I2S-R microphone
- Breadboard
- MATLAB script to check impulse response measurements

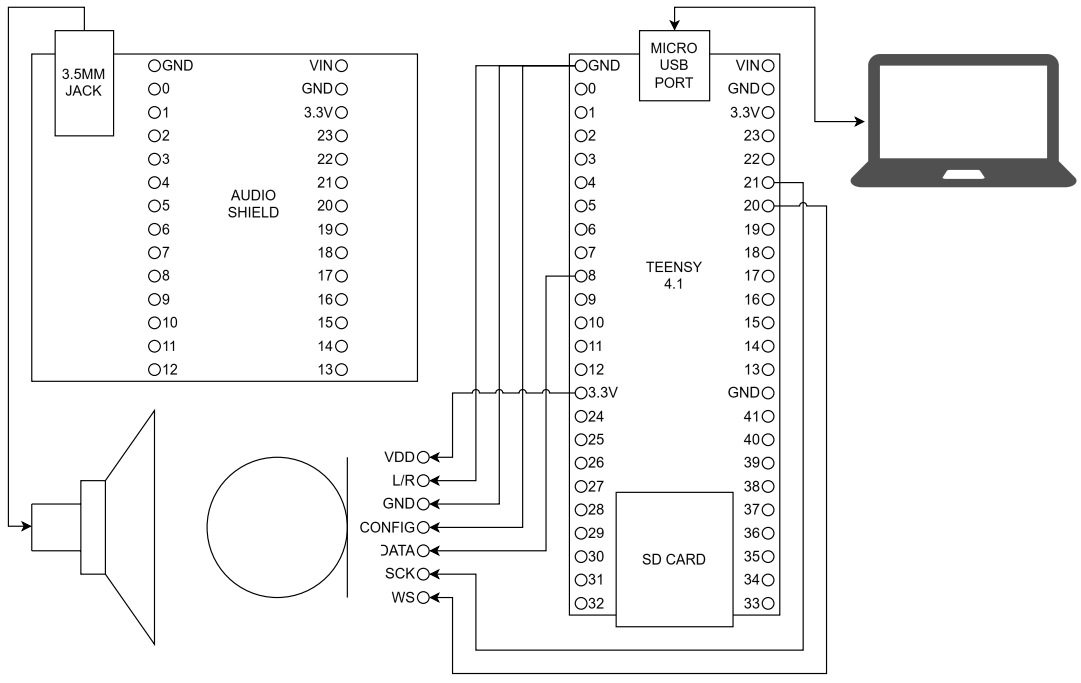


Figure 5.17: Setup for testing the microphone with a known signal. The audio shield utilizes the same pins as the Teensy.

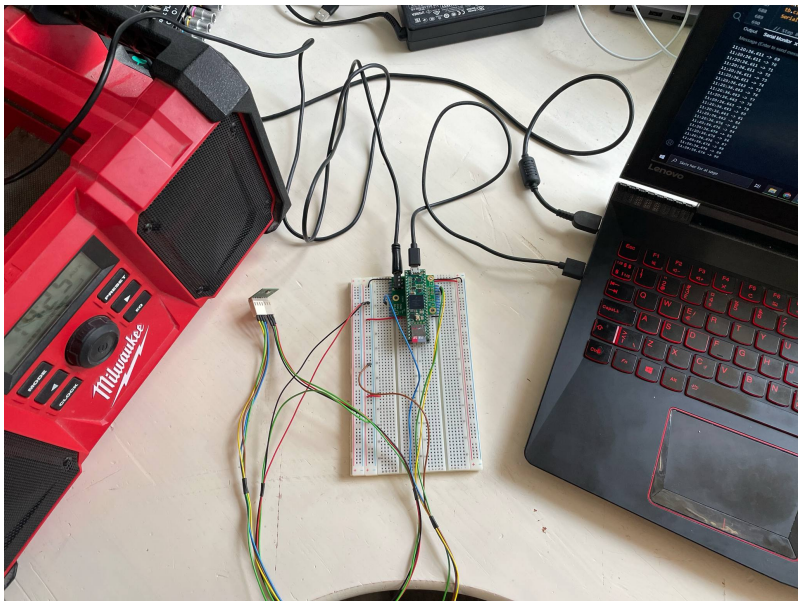


Figure 5.18: Real life test setup for FFT and impulse response measurements.

Actual Testing Testing showed rather quickly that the initial impulse response measurement led to corrupted or otherwise faulty data. Therefore, the second implementation mentioned in the design was developed and proved functional.

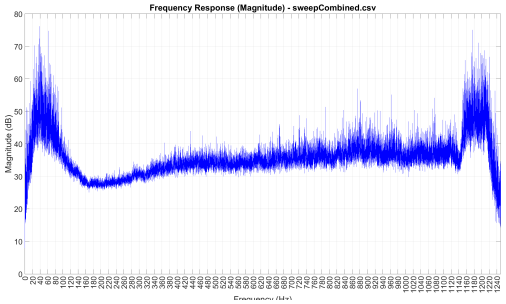
Test Results The real-time FFT reading shows that the teensy is capable of discerning frequencies into 20 Hz wide bins using the standard FFT library, shown in figure 5.19. Unfortunately, bleeding can be seen in the adjacent bins as well, suggesting that either the microphone, the FFT algorithm, or the sine generator/speaker combo is not completely pure.

```
12:06:32.685 -> 400.0-420.0: 0.024170  420.0-440.0: 0.041992  440.0-460.0: 0.018250  460.0-480.0: 0.000977  480.0-500.0: 0.001404
12:06:32.685 -> 400.0-420.0: 0.025085  420.0-440.0: 0.041809  440.0-460.0: 0.017151  460.0-480.0: 0.001526  480.0-500.0: 0.001221
12:06:32.721 -> 400.0-420.0: 0.024597  420.0-440.0: 0.041931  440.0-460.0: 0.018555  460.0-480.0: 0.000488  480.0-500.0: 0.001160
12:06:32.721 -> 400.0-420.0: 0.024475  420.0-440.0: 0.042419  440.0-460.0: 0.017761  460.0-480.0: 0.000854  480.0-500.0: 0.000793
12:06:32.721 -> 400.0-420.0: 0.025879  420.0-440.0: 0.042603  440.0-460.0: 0.017639  460.0-480.0: 0.001587  480.0-500.0: 0.000305
12:06:32.721 -> 400.0-420.0: 0.022644  420.0-440.0: 0.041687  440.0-460.0: 0.018372  460.0-480.0: 0.000183  480.0-500.0: 0.000244
12:06:32.756 -> 400.0-420.0: 0.023560  420.0-440.0: 0.041382  440.0-460.0: 0.018677  460.0-480.0: 0.000793  480.0-500.0: 0.000671
12:06:32.756 -> 400.0-420.0: 0.023865  420.0-440.0: 0.040771  440.0-460.0: 0.017029  460.0-480.0: 0.000244  480.0-500.0: 0.000854
12:06:32.756 -> 400.0-420.0: 0.025391  420.0-440.0: 0.042053  440.0-460.0: 0.019043  460.0-480.0: 0.001038  480.0-500.0: 0.001465
12:06:32.791 -> 400.0-420.0: 0.023743  420.0-440.0: 0.042297  440.0-460.0: 0.018616  460.0-480.0: 0.000977  480.0-500.0: 0.000977
12:06:32.791 -> 400.0-420.0: 0.024170  420.0-440.0: 0.042908  440.0-460.0: 0.018982  460.0-480.0: 0.000122  480.0-500.0: 0.000854
12:06:32.791 -> 400.0-420.0: 0.024597  420.0-440.0: 0.042114  440.0-460.0: 0.017700  460.0-480.0: 0.000488  480.0-500.0: 0.000366
12:06:32.828 -> 400.0-420.0: 0.023621  420.0-440.0: 0.041504  440.0-460.0: 0.018799  460.0-480.0: 0.000732  480.0-500.0: 0.000793
12:06:32.828 -> 400.0-420.0: 0.024963  420.0-440.0: 0.041138  440.0-460.0: 0.017944  460.0-480.0: 0.000488  480.0-500.0: 0.000671
12:06:32.828 -> 400.0-420.0: 0.023010  420.0-440.0: 0.039673  440.0-460.0: 0.016296  460.0-480.0: 0.001160  480.0-500.0: 0.000549
12:06:32.860 -> 400.0-420.0: 0.020081  420.0-440.0: 0.035950  440.0-460.0: 0.015991  460.0-480.0: 0.001160  480.0-500.0: 0.000854
12:06:32.860 -> 400.0-420.0: 0.017883  420.0-440.0: 0.032532  440.0-460.0: 0.013428  460.0-480.0: 0.001038  480.0-500.0: 0.000977
12:06:32.860 -> 400.0-420.0: 0.018921  420.0-440.0: 0.031921  440.0-460.0: 0.013245  460.0-480.0: 0.001892  480.0-500.0: 0.001953
12:06:32.896 -> 400.0-420.0: 0.017944  420.0-440.0: 0.031372  440.0-460.0: 0.012329  460.0-480.0: 0.000916  480.0-500.0: 0.002075
12:06:32.896 -> 400.0-420.0: 0.016296  420.0-440.0: 0.029175  440.0-460.0: 0.014587  460.0-480.0: 0.002197  480.0-500.0: 0.002563
12:06:32.896 -> 400.0-420.0: 0.012512  420.0-440.0: 0.024231  440.0-460.0: 0.011597  460.0-480.0: 0.001770  480.0-500.0: 0.003723
12:06:32.931 -> 400.0-420.0: 0.015869  420.0-440.0: 0.022766  440.0-460.0: 0.008423  460.0-480.0: 0.002014  480.0-500.0: 0.001465
12:06:32.931 -> 400.0-420.0: 0.011841  420.0-440.0: 0.019287  440.0-460.0: 0.008972  460.0-480.0: 0.002747  480.0-500.0: 0.002991
12:06:32.931 -> 400.0-420.0: 0.009094  420.0-440.0: 0.015625  440.0-460.0: 0.006226  460.0-480.0: 0.000427  480.0-500.0: 0.001160
12:06:32.965 -> 400.0-420.0: 0.006653  420.0-440.0: 0.011902  440.0-460.0: 0.004944  460.0-480.0: 0.001099  480.0-500.0: 0.001465
12:06:32.965 -> 400.0-420.0: 0.004089  420.0-440.0: 0.010559  440.0-460.0: 0.005249  460.0-480.0: 0.001038  480.0-500.0: 0.000610
12:06:33.000 -> 400.0-420.0: 0.005798  420.0-440.0: 0.007507  440.0-460.0: 0.002563  460.0-480.0: 0.000854  480.0-500.0: 0.001831
12:06:33.000 -> 400.0-420.0: 0.004150  420.0-440.0: 0.006226  440.0-460.0: 0.002625  460.0-480.0: 0.000244  480.0-500.0: 0.000671
12:06:33.000 -> 400.0-420.0: 0.003540  420.0-440.0: 0.003906  440.0-460.0: 0.003479  460.0-480.0: 0.003296  480.0-500.0: 0.003052
12:06:33.000 -> 400.0-420.0: 0.001953  420.0-440.0: 0.002258  440.0-460.0: 0.000183  460.0-480.0: 0.001038  480.0-500.0: 0.000671
12:06:33.034 -> 400.0-420.0: 0.000366  420.0-440.0: 0.001587  440.0-460.0: 0.000854  460.0-480.0: 0.000427  480.0-500.0: 0.000305
12:06:33.034 -> 400.0-420.0: 0.000854  420.0-440.0: 0.001282  440.0-460.0: 0.000549  460.0-480.0: 0.000488  480.0-500.0: 0.000183
12:06:33.034 -> 400.0-420.0: 0.000305  420.0-440.0: 0.000977  440.0-460.0: 0.000732  460.0-480.0: 0.000305  480.0-500.0: 0.000488
12:06:33.069 -> 400.0-420.0: 0.000732  420.0-440.0: 0.000427  440.0-460.0: 0.000244  460.0-480.0: 0.000366  480.0-500.0: 0.000610
12:06:33.069 -> 400.0-420.0: 0.000305  420.0-440.0: 0.000610  440.0-460.0: 0.000916  460.0-480.0: 0.000916  480.0-500.0: 0.000427
12:06:33.069 -> 400.0-420.0: 0.000610  420.0-440.0: 0.000427  440.0-460.0: 0.000183  460.0-480.0: 0.000244  480.0-500.0: 0.000549
12:06:33.103 -> 400.0-420.0: 0.000427  420.0-440.0: 0.000244  440.0-460.0: 0.000549  460.0-480.0: 0.000183  480.0-500.0: 0.000366
12:06:33.103 -> 400.0-420.0: 0.001099  420.0-440.0: 0.000977  440.0-460.0: 0.001160  460.0-480.0: 0.000732  480.0-500.0: 0.000977
```

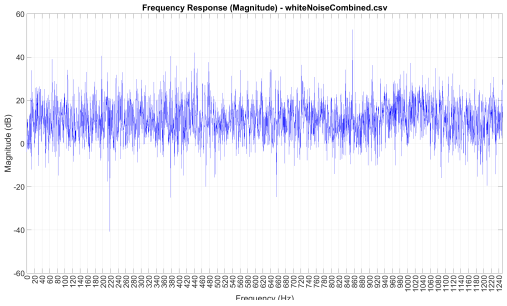
Figure 5.19: FFT of microphone subjected to 450Hz sine for the first half, and background noise in the second half.

Testing the impulse response yielded highly favorable results, as can be seen in figure 5.20. In figure 5.20a, it is evident that the frequency response in the grand scheme is flat, except around 20-80Hz and 1120-1220Hz. The low-frequency issues could be explained by the frequency response for the PUI Audio DMM-4026-B-I2S-R microphone itself, shown in figure 5.21. Nonetheless, that should only account for the 20-30Hz range. The remaining deviances are thought to be artifacts introduced by the "Milwaukee M18 JSR" construction site speaker that has been used. However, something peculiar happens if the sweep is increased from 20-20kHz, as the "highfrequency" artifacts disappear completely when zoomed in on 20-1250Hz, as shown in figure 5.22.

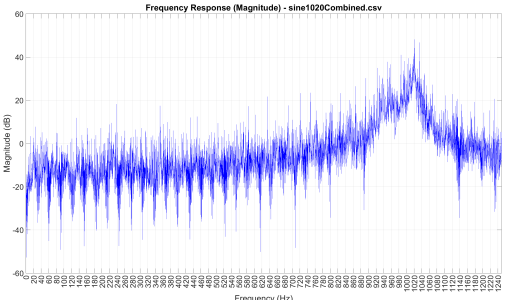
In appendix "Frequency Domain Plots 1 Microphone" enlarged images of each frequency response can be seen, and starting page 191 in figure K.1 plots from 0-20kHz can be seen, where a sweep from 20Hz to 20kHz is also found. This sweep shows that either the speaker or the microphone has a substantial error, as the sweep exhibits extreme amounts of noise.



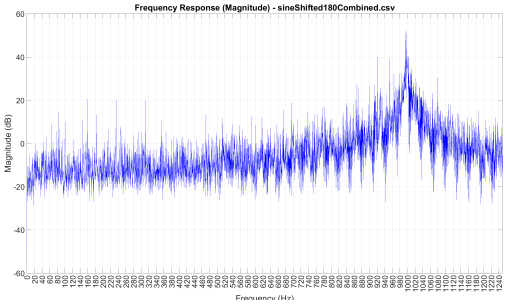
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.



(c) 1020Hz puretone sinusoid frequency domain impulse response.



(d) 1kHz phase shifted sinusoid frequency domain impulse response.

Figure 5.20: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy. All plots span 0-1250Hz.

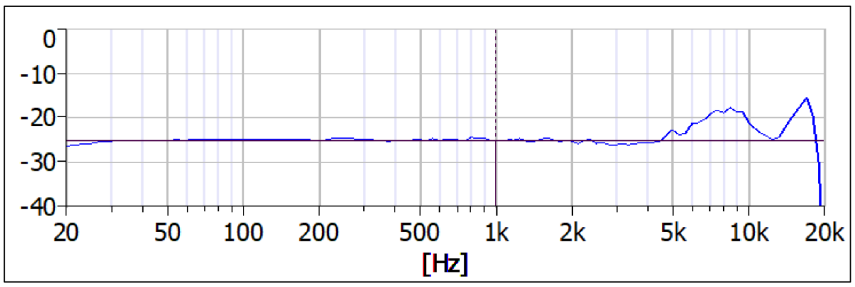


Figure 5.21: Typical frequency response of the PUI Audio DMM-4026-B-I2S-R microphone, [68].

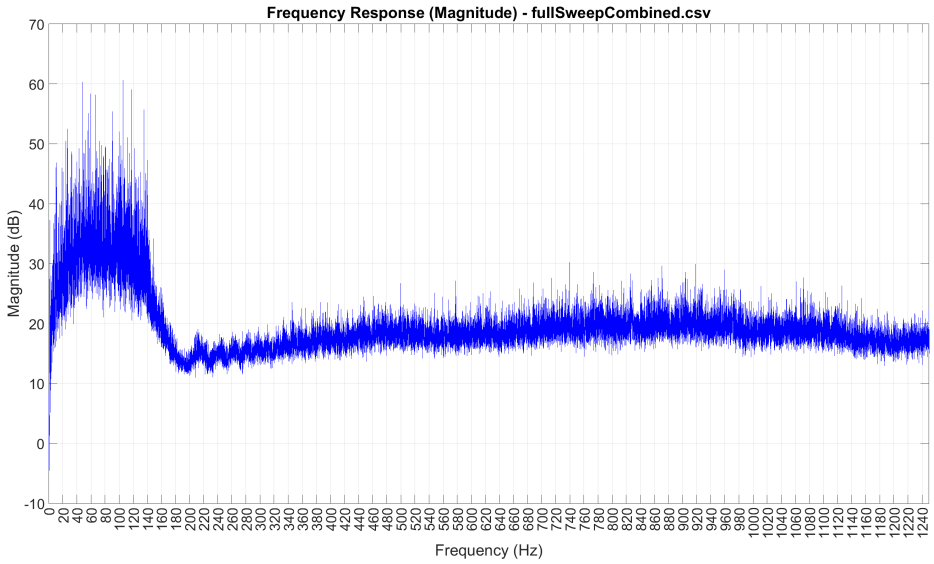


Figure 5.22: Sweep from 20Hz to 20kHz, with plot focused on the band from 20Hz to 1250Hz.

Summary

In general the Teensy has proven capable of performing realtime FFT analysis, but not as exact as desired. Luckily, it has instead proven to be fully capable of measuring an impulse response, and saving it to an SD card in .CSV format. This enabled further analysis to be made in MATLAB.

5.3.2 Measuring With 2 Microphones

To create an impedance tube capable measuring acoustic properties with the transfer matrix method, at least two microphones must be functional at a time. Knowledge and introductional code from section 5.3.1 will be reused to ease development.

Design

Most of the design from section 5.3.1 will be identical to this, and is therefore not discussed. The only major design changes are the addition of a microphone on the right channel on the I2S line used for FFT and impulse response measurement.

Implementation

Implemention the FFT is simply done by altering the first few lines shown in section 5.3.1 to:

```
AudioInputI2S            i2sMic;
AudioAnalyzeFFT1024      fft10241;
AudioConnection          patchCord1(i2sMic, 0, fft10241, 0); // Set to
    channel 1
AudioAnalyzeFFT1024      fft10242;
AudioConnection          patchCord2(i2sMic, 1, fft10242, 0); // Set to
    channel 2
```

As the program is now using both microphones. Physically on the microphones, one must have the L/R cable plugged into VDD, and the other into GND, as this is how they designates L/R. In the setup function:

```
fft1024.windowFunction(AudioWindowRectangular1024);
```

is replaced by:

```
fft10241.windowFunction(AudioWindowRectangular1024);
fft10242.windowFunction(AudioWindowRectangular1024);
```

And in the loop function, the if statement is altered similarly, to match "fft10241" or "fft10242".

The impulse response measurement is similarly simple to alter 2 microphone accomodation into, as another AudioRecordQueue is added, and the "recordBothToFileSingleFile" is updated with a third buffer, saving the third signal to the SD card:

```
AudioRecordQueue mic1Queue;  // Recordqueue for the 1st microphone
AudioConnection patchCord8(i2sMicSet1, 0, mic1Queue, 0);
AudioRecordQueue mic2Queue;  // Recordqueue for the 2nd microphone
AudioConnection patchCord9(i2sMicSet1, 1, mic2Queue, 0);

void recordThreeToFileSingleFile(File& file, uint32_t totalSamples) {
  uint32_t samplesRecorded = 0;                                          //
    Checker for when to stop
  while (samplesRecorded < totalSamples) {
    if (recordQueue.available() && mic1Queue.available()) {  // Checks for
    data on all signal lines, ensuring that data is present on all lines
    simultaneously
      int16_t* bufMixer = recordQueue.readBuffer();
      int16_t* bufMic1 = mic1Queue.readBuffer();
      int16_t* bufMic2 = mic2Queue.readBuffer();
      for (int i = 0; i < BLOCK_SIZE; i++) {  // Divides signals into
    blocks to ensure no overflow is happening
        float sMixer = (float)bufMixer[i] / 32768.0f;
        float sMic1 = (float)bufMic1[i] / 32768.0f;
        float sMic2 = (float)bufMic2[i] / 32768.0f;
        file.print(sMixer, 6);  // Saves mixer signal to SD card
        file.print(",");        // Saves to SD card
        file.print(sMic1, 6);   // Saves mic 1 signal to SD card
        file.print(",");        // Saves to SD card
        file.print(sMic2, 6);   // Saves mic2 signal to SD card
        file.print("\n");
      }
      recordQueue.freeBuffer();       // Frees buffer
      mic1Queue.freeBuffer();         // Frees buffer
      mic2Queue.freeBuffer();         // Frees buffer
      samplesRecorded += BLOCK_SIZE;  // Keeps track of how many samples
    has been saved
    }
  }
}
```

Full code implementation can be found on GitHub here or in the attached files as "XXXX".

Test

The purpose of the test is to ensure that the Teensy is capable of measuring FFTs and impulse responses from 2 microphones at the same time.

Test Setup In general, the setup will be identical to the one shown in section 5.3.1, with the exception of an added microphone, as shown in figure 5.23. Everything else is exactly the same.

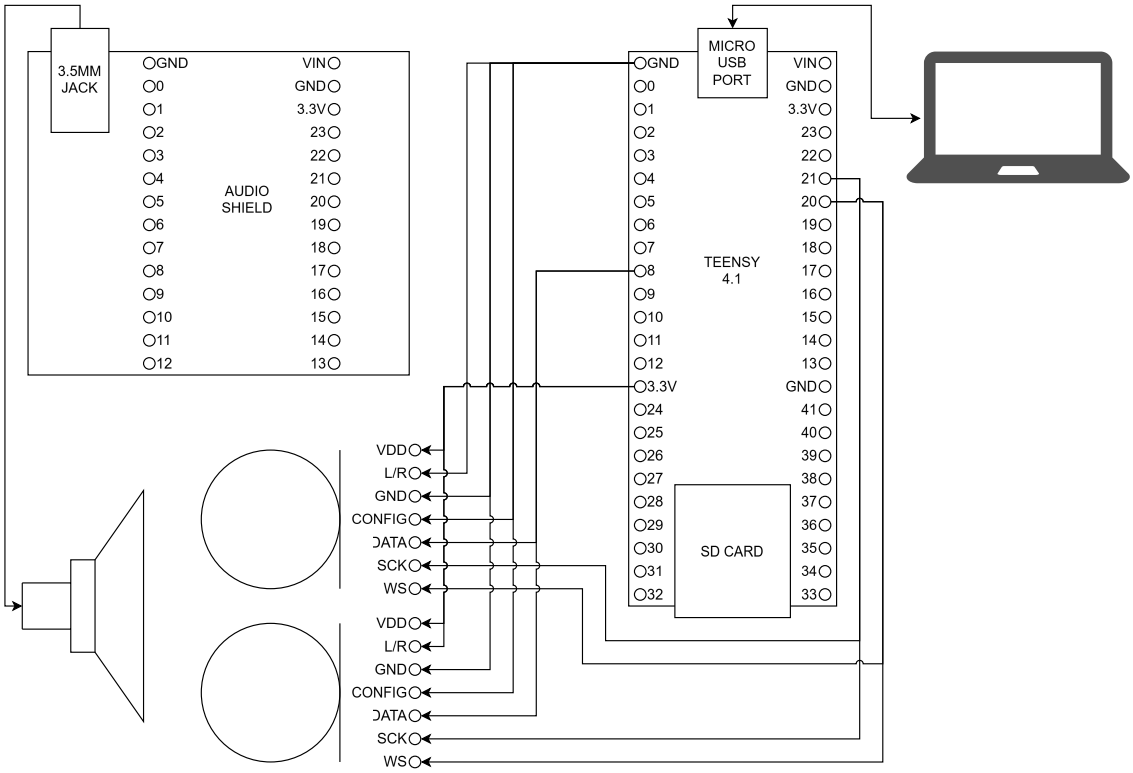


Figure 5.23: Setup for testing 2 microphones with a known signal. The audio shield utilizes the same pins as the Teensy. Notice that the only difference in adding a second mic is the L/R cables' position.

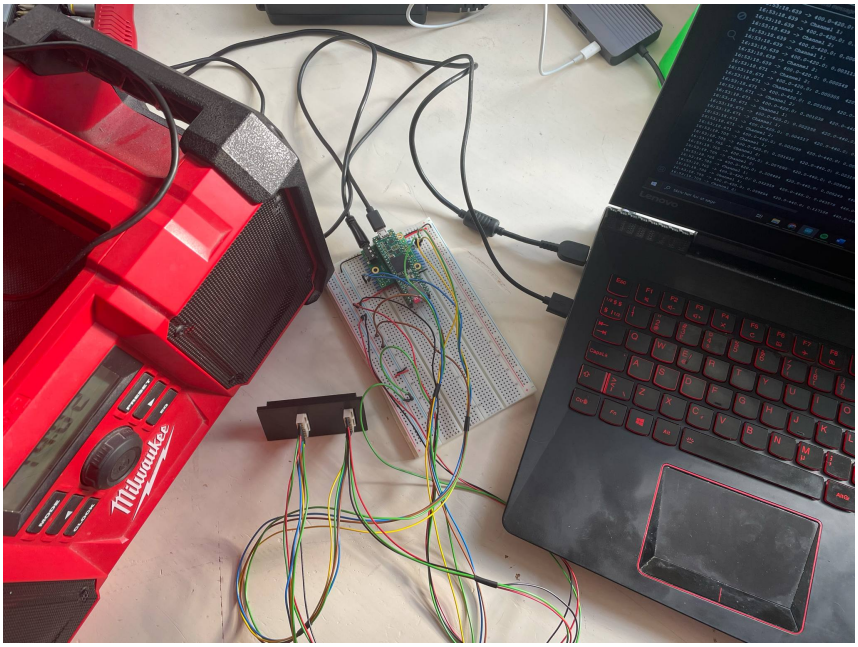
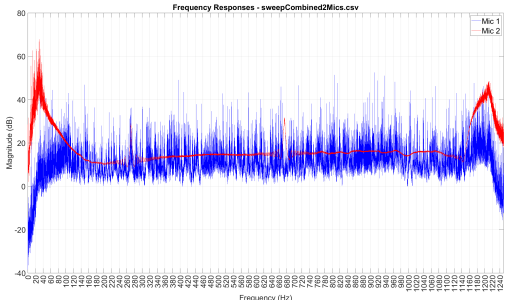
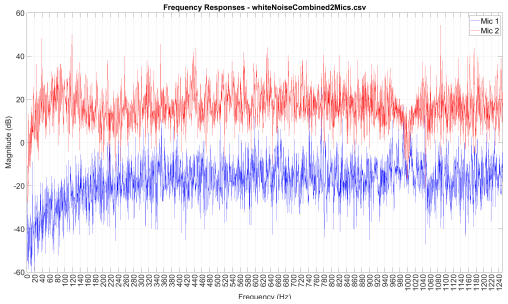


Figure 5.24: Real life test setup for FFT and impulse response measurements with 2 microphones.

Actual Testing During the very first test, it became apparent that the two microphones created vastly different impulse responses, which can be seen in figure 5.25, suggesting that the microphones themselves had significant differences in their impulse responses. These differences were therefore tested, by measuring individual impulse responses for each microphone, which can be found in appendix Frequency Domain Plots Individual Microphones starting page 182.



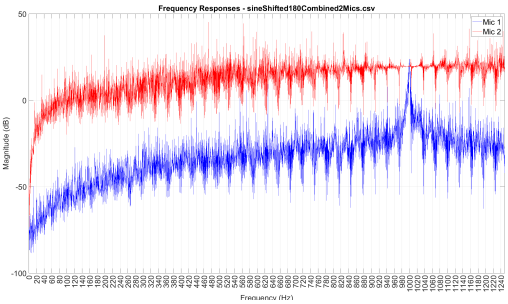
(a) Sine sweep from 20-1220Hz performed with two randomly chosen microphones.



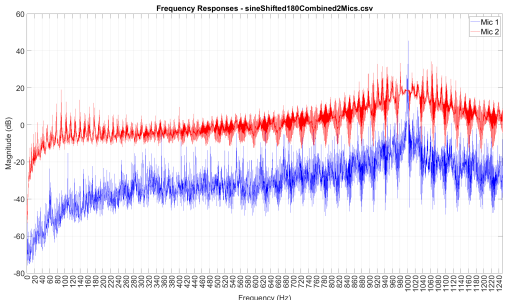
(b) White noise with two randomly chosen microphones.

Figure 5.25: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy first time around with randomly chosen microphones. Note the large difference in amplitude between them and the deviating impulse responses.

After having ranked each and every microphone's impulse response, the two most proficient microphones (1 and 8) were chosen for further testing. The experiment was redone, but results similar to the first were achieved. In figure 5.26, frequency responses for the 1kHz phase shift test can be seen. In the reiteration of the test, the microphones are connected to the same I2S channel, with the only difference between them being their L/R assignment. Despite being connected to the same data/power lines, the microphone connected to R (mic 2) consistently shows a frequency response approximately 40 dB higher than the L channel (mic 1). Despite switching which microphone is attached to L/R, the result remains similar.



(a) 1kHz puretone sine being played to both microphones. Mic 1 in this plot is microphone 1, and mic 2 is microphone 8..



(b) 1kHz puretone sine being played to both microphones. Mic 1 in this plot is microphone 8, and mic 2 is microphone 1.

Figure 5.26: Frequency domain plots of 1kHz puretone sinusoid. The only difference made between the two plots are L/R orientation, the rest of the circuitry is exactly the same.

To mitigate this large difference, the second I2S channel is utilized. To use the second I2S channel, the physical connections are changed to match figure 5.27, and the code setup has to be refactored into:

```
AudioInputI2S i2sMic1;
AudioInputI2S2 i2sMic2; // Initialises I2S2
AudioRecordQueue mic1Queue;  // Recordqueue for the 1st microphone
AudioConnection patchCord8(i2sMic1, 0, mic1Queue, 0);
AudioRecordQueue mic2Queue;  // Recordqueue for the 2nd microphone
AudioConnection patchCord9(i2sMic2, 0, mic2Queue, 0);
```

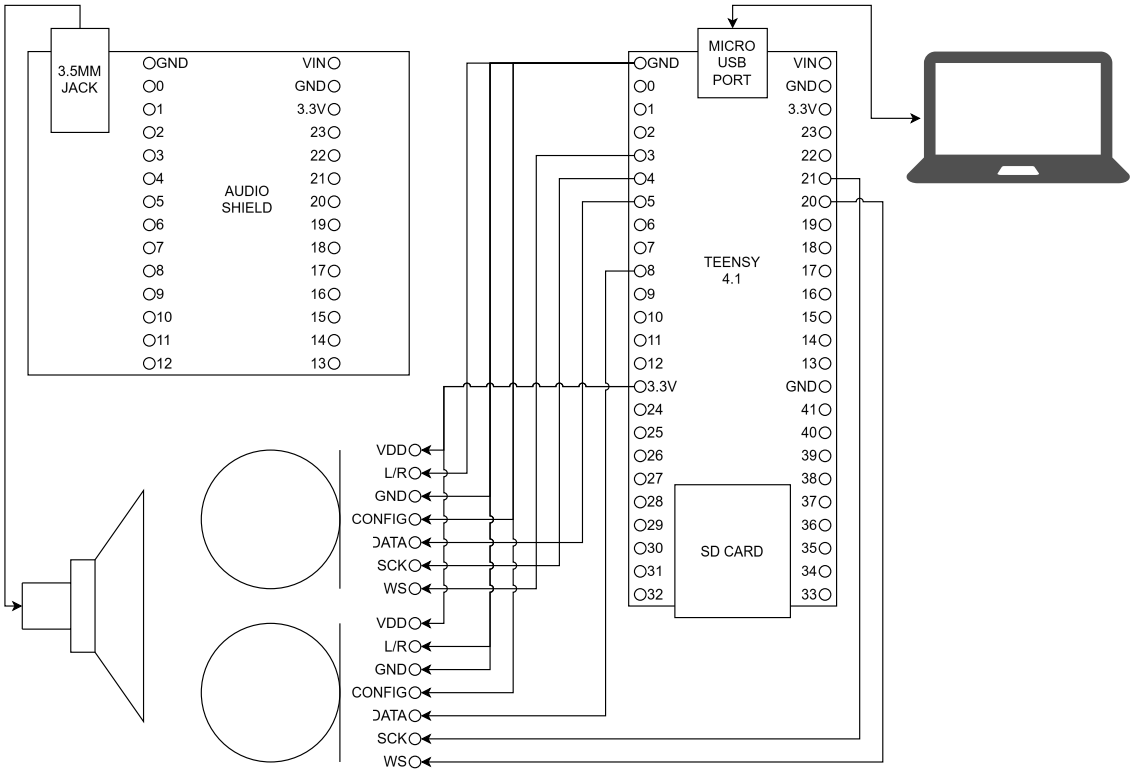


Figure 5.27: Schematic for measuring with 2 microphones on two different I2S channels.

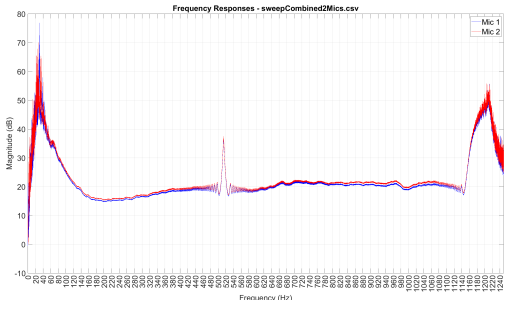
Nevertheless, this change in software and hardware brought no results. The corresponding impulse responses remained similar. It was not until after having tried several different code iterations, hardware modifications, and microphone combinations that the fault was found. In the if loop on line 9 in the first code snippet, the if statement doesn't care if the second microphone is done writing data or not, resulting in only the very first few bits being written, resulting in a minimal fractional value. The code was then refactored to the following, and a single I2S channel was used again:

```
1 if (recordQueue.available() && mic1Queue.available()) {  // FAULTY
2 if (recordQueue.available() && mic1Queue.available() && mic2Queue.available
    ()) { // FUNCTIONAL
```

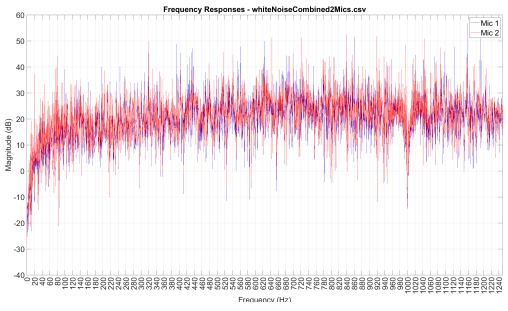
Test Results As the software was now functional, proper results could be achieved. In figure 5.28, a serial output from the rectangular window can be seen. More interesting is figure 5.29, as this is the impulse responses for two microphones subjected to the same signals. Differences in the signals are thought to be the individuality of the microphones, as they are by no means calibrated. Nevertheless, the signals are very much alike, proving that the Teensy is capable of such data obtainment. The noise present on all signals apart from the sweep is thought to be caused by the rather short signal length used, as the sweep is set to 10 seconds, whereas all other signals only get 1 second of playtime. However, testing this hypothesis led to the discovery that besides the fault in the if loop mentioned earlier, the cables between the Teensy and microphones also had a part in the wrongful data. Despite best efforts being made in soldering, crimping, and sealing all cables, data transfer is either unstable or being affected by noise, as can be seen in figure 5.30. All frequency and time domain plots can be found in large scale format in appendix "Frequency and Time Domain Plots 2 Microphones" starting page 201.

```
16:53:18.671 -> Channel 1:
16:53:18.671 -> 400.0-420.0: 0.001038  420.0-440.0: 0.001953  440.0-460.0: 0.003296  460.0-480.0: 0.002319  480.0-500.0: 0.001221
16:53:18.671 -> Channel 2:
16:53:18.671 -> 400.0-420.0: 0.001038  420.0-440.0: 0.003052  440.0-460.0: 0.003357  460.0-480.0: 0.000977  480.0-500.0: 0.001831
16:53:18.671 -> Channel 1:
16:53:18.671 -> 400.0-420.0: 0.002258  420.0-440.0: 0.002930  440.0-460.0: 0.003845  460.0-480.0: 0.003174  480.0-500.0: 0.003784
16:53:18.671 -> Channel 1:
16:53:18.671 -> 400.0-420.0: 0.004517  420.0-440.0: 0.004700  440.0-460.0: 0.009705  460.0-480.0: 0.004333  480.0-500.0: 0.004761
16:53:18.671 -> Channel 2:
16:53:18.671 -> 400.0-420.0: 0.002258  420.0-440.0: 0.005310  440.0-460.0: 0.004578  460.0-480.0: 0.001587  480.0-500.0: 0.000732
16:53:18.705 -> Channel 1:
16:53:18.705 -> 400.0-420.0: 0.031616  420.0-440.0: 0.023193  440.0-460.0: 0.019043  460.0-480.0: 0.027527  480.0-500.0: 0.026489
16:53:18.705 -> Channel 2:
16:53:18.705 -> 400.0-420.0: 0.007690  420.0-440.0: 0.009949  440.0-460.0: 0.007813  460.0-480.0: 0.000977  480.0-500.0: 0.005066
16:53:18.705 -> Channel 2:
16:53:18.705 -> 400.0-420.0: 0.008484  420.0-440.0: 0.005249  440.0-460.0: 0.005310  460.0-480.0: 0.006165  480.0-500.0: 0.003845
16:53:18.705 -> Channel 1:
16:53:18.705 -> 400.0-420.0: 0.032959  420.0-440.0: 0.043579  440.0-460.0: 0.048035  460.0-480.0: 0.028259  480.0-500.0: 0.014465
16:53:18.705 -> Channel 1:
16:53:18.705 -> 400.0-420.0: 0.099854  420.0-440.0: 0.117188  440.0-460.0: 0.080933  460.0-480.0: 0.046387  480.0-500.0: 0.037476
16:53:18.705 -> Channel 2:
16:53:18.705 -> 400.0-420.0: 0.012817  420.0-440.0: 0.013306  440.0-460.0: 0.010620  460.0-480.0: 0.004761  480.0-500.0: 0.003479
16:53:18.740 -> Channel 1:
16:53:18.740 -> 400.0-420.0: 0.077148  420.0-440.0: 0.098633  440.0-460.0: 0.076294  460.0-480.0: 0.040344  480.0-500.0: 0.029785
16:53:18.740 -> Channel 2:
16:53:18.740 -> 400.0-420.0: 0.013916  420.0-440.0: 0.026672  440.0-460.0: 0.018066  460.0-480.0: 0.005981  480.0-500.0: 0.009277
16:53:18.740 -> Channel 1:
16:53:18.740 -> 400.0-420.0: 0.099182  420.0-440.0: 0.111938  440.0-460.0: 0.081604  460.0-480.0: 0.028687  480.0-500.0: 0.012512
16:53:18.740 -> Channel 2:
16:53:18.740 -> 400.0-420.0: 0.012573  420.0-440.0: 0.029236  440.0-460.0: 0.021240  460.0-480.0: 0.006836  480.0-500.0: 0.013672
16:53:18.740 -> Channel 1:
16:53:18.740 -> 400.0-420.0: 0.026978  420.0-440.0: 0.021484  440.0-460.0: 0.033020  460.0-480.0: 0.018005  480.0-500.0: 0.010254
16:53:18.740 -> Channel 2:
16:53:18.740 -> 400.0-420.0: 0.021790  420.0-440.0: 0.017273  440.0-460.0: 0.011902  460.0-480.0: 0.012024  480.0-500.0: 0.010559
16:53:18.775 -> Channel 1:
16:53:18.775 -> 400.0-420.0: 0.093689  420.0-440.0: 0.103760  440.0-460.0: 0.062622  460.0-480.0: 0.020203  480.0-500.0: 0.020386
16:53:18.775 -> Channel 2:
16:53:18.775 -> 400.0-420.0: 0.023193  420.0-440.0: 0.016724  440.0-460.0: 0.011475  460.0-480.0: 0.010071  480.0-500.0: 0.004211
16:53:18.775 -> Channel 1:
16:53:18.775 -> 400.0-420.0: 0.134155  420.0-440.0: 0.163452  440.0-460.0: 0.111206  460.0-480.0: 0.038513  480.0-500.0: 0.045776
16:53:18.775 -> Channel 2:
16:53:18.775 -> 400.0-420.0: 0.013062  420.0-440.0: 0.036133  440.0-460.0: 0.032043  460.0-480.0: 0.011292  480.0-500.0: 0.018555
16:53:18.775 -> Channel 1:
16:53:18.814 -> 400.0-420.0: 0.136475  420.0-440.0: 0.178284  440.0-460.0: 0.127930  460.0-480.0: 0.031494  480.0-500.0: 0.032410
16:53:18.814 -> Channel 2:
16:53:18.814 -> 400.0-420.0: 0.034363  420.0-440.0: 0.052673  440.0-460.0: 0.026306  460.0-480.0: 0.010681  480.0-500.0: 0.017883
```

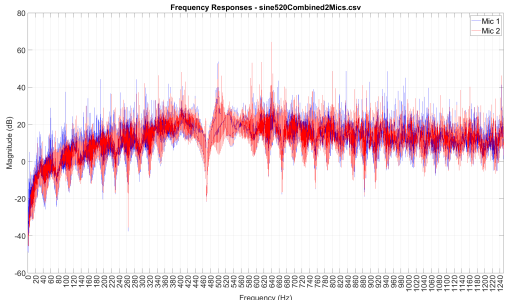
Figure 5.28: FFT output measured by both channels/microphones. At first, nothing but background noise until a sine wave at 450Hz is played.



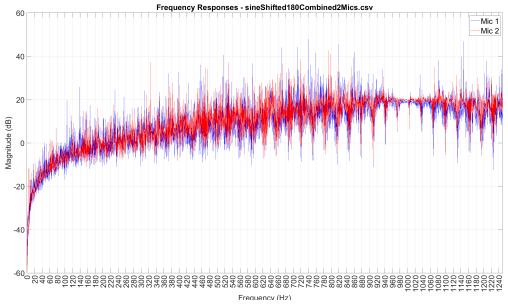
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.



(c) 520Hz puretone sinusoid frequency domain impulse response.



(d) 1kHz phase shifted sinusoid frequency domain impulse response.

Figure 5.29: Plots of frequency domain impulse responses derived from the impulse responses measured by 2 microphones on 1 I2S channel on the Teensy. All plots span 0-1250Hz.

```
22:53:43.268 -> CH1[0]: 730     CH2[0]: 666
22:53:43.268 -> CH1[0]: 699     CH2[0]: 644
22:53:43.300 -> CH1[0]: 708     CH2[0]: 673
22:53:43.300 -> CH1[0]: 662     CH2[0]: 654
22:53:43.300 -> CH1[0]: 681     CH2[0]: 665
22:53:43.300 -> CH1[0]: 662     CH2[0]: 649
22:53:43.300 -> CH1[0]: 638     CH2[0]: 710
22:53:43.300 -> CH1[0]: 575     CH2[0]: 645
22:53:43.300 -> CH1[0]: 571     CH2[0]: 661
22:53:43.300 -> CH1[0]: 685     CH2[0]: 646
22:53:43.300 -> CH1[0]: 665     CH2[0]: 704
22:53:43.300 -> CH1[0]: 580     CH2[0]: 688
22:53:43.300 -> CH1[0]: 537     CH2[0]: 640
```

(a) Stable output derived from cables lying still.

```
22:53:43.492 -> CH1[0]: 287     CH2[0]: 583
22:53:43.492 -> CH1[0]: 0       CH2[0]: 576
22:53:43.492 -> CH1[0]: 24432   CH2[0]: 612
22:53:43.492 -> CH1[0]: 4032    CH2[0]: 595
22:53:43.492 -> CH1[0]: 3952    CH2[0]: 545
22:53:43.492 -> CH1[0]: 25495   CH2[0]: 621
22:53:43.492 -> CH1[0]: 10272   CH2[0]: 674
22:53:43.492 -> CH1[0]: 416     CH2[0]: 570
22:53:43.492 -> CH1[0]: 8720    CH2[0]: 534
22:53:43.492 -> CH1[0]: 2341    CH2[0]: 663
22:53:43.492 -> CH1[0]: 5354    CH2[0]: 654
22:53:43.524 -> CH1[0]: 7916    CH2[0]: 563
22:53:43.524 -> CH1[0]: -32664  CH2[0]: 579
```

(b) Unstable output derived from fidgeting the cables

Figure 5.30: Serial monitor output showing clearly how measurements can become deranged if cables are not lying perfectly.

Summary

Implementation and testing of 2 microphones at once has yielded satisfactory results, and further development can ensue. Microphones show near-identical frequency responses when cables are lying perfectly. When cables are somehow twisted or otherwise imperfectly placed, the results are of very poor quality. The obvious solution to mitigate this problem would be to rewire each microphone or to place them in a perfboard. Unfortunately, such physical alterations are not feasible to implement at the current point of the project.

5.3.3 Measuring With 6 Microphones

As the physical tube enables the possibility of multiple microphone sets being used at once, it would be practical to implement software capable of measuring all 3 microphone sets at once, enabling the user to create averaged values across all three transfer matrices. This is beneficial, as none of the 3 microphone sets hit every physical design aspect recommended in the DS 10534:2023 standard, but each set correspond to a multitude of the recommendations. This is explained in depth in section 5.1.6 starting page 56.

Design

To utilize acquired knowledge efficiently, previous design choices from section 5.3.1 is used once again. However, some changes are applied, as the "input_i2s_hex.h" library is used. The general idea of this library is to utilize a time division multiplexing (TDM) based scheduling, which will enable up to 8 microphones on each I2S channel (if "input_i2s_octo.h were used instead). By utilizing the hex library, all 6 microphones can be connected to the same I2S channel, minimizing the possibility of timing errors.

Implementation

As before, most of section 5.3.1 can be reused. The setup for an FFT approach would be:

```
1 AudioInputI2SHex i2sHex1;
2 AudioAnalyzeFFT1024 fft1024CH1;
3 -||-
4 AudioAnalyzeFFT1024 fft1024CH6;
5 AudioConnection patchCord1(i2sHex1, 0, fft1024CH1, 0);
```

```
-||-
AudioConnection patchCord6(i2sHex1, 5, fft1024CH6, 0);
fft10241.windowFunction(AudioWindowRectangular1024);
-||-
fft10246.windowFunction(AudioWindowRectangular1024);
```

And the impulse response implementation is:

```
AudioRecordQueue mic1Queue;  // Recordqueue for the 1st microphone
AudioConnection patchCord8(i2sHex1, 0, mic1Queue, 0);
-||-
AudioRecordQueue mic6Queue;  // Recordqueue for the 2nd microphone
AudioConnection patchCord14(i2sHext1, 5, mic6Queue, 0);

void recordThreeToFileSingleFile(File& file, uint32_t totalSamples) {
  uint32_t samplesRecorded = 0;                                          //
    Checker for when to stop
  while (samplesRecorded < totalSamples) {
    if (recordQueue.available() && mic1Queue.available()) {  // Checks for
    data on all signal lines, ensuring that data is present on all lines
    simultaneously
      int16_t* bufMixer = recordQueue.readBuffer();
      int16_t* bufMic1 = mic1Queue.readBuffer();
      -||-
      int16_t* bufMic6 = mic6Queue.readBuffer();
      for (int i = 0; i < BLOCK_SIZE; i++) {  // Divides signals into
    blocks to ensure no overflow is happening
        float sMixer = (float)bufMixer[i] / 32768.0f;
        float sMic1 = (float)bufMic1[i] / 32768.0f;
      -||-
        float sMic2 = (float)bufMic6[i] / 32768.0f;
        file.print(sMixer, 6);  // Saves mixer signal to SD card
        file.print(",");        // Saves to SD card
        file.print(sMic1, 6);   // Saves mic 1 signal to SD card
      -||-
        file.print(",");        // Saves to SD card
        file.print(sMic6, 6);   // Saves mic2 signal to SD card
        file.print("\n");
      }
      recordQueue.freeBuffer();       // Frees buffer
      mic1Queue.freeBuffer();         // Frees buffer
      mic2Queue.freeBuffer();         // Frees buffer
      samplesRecorded += BLOCK_SIZE;  // Keeps track of how many samples
    has been saved
    }
  }
}
```

Test

The test's purpose is to confirm that the PUI DMM 4026-B-I2S-R can function with the hex library and thereby enable the system to measure all 6 input signals on a single I2S channel.

Test Setup Testing 6 microphones at once will be almost identical to all previous tests. The main difference will be the microphones attached to the same I2S channel.

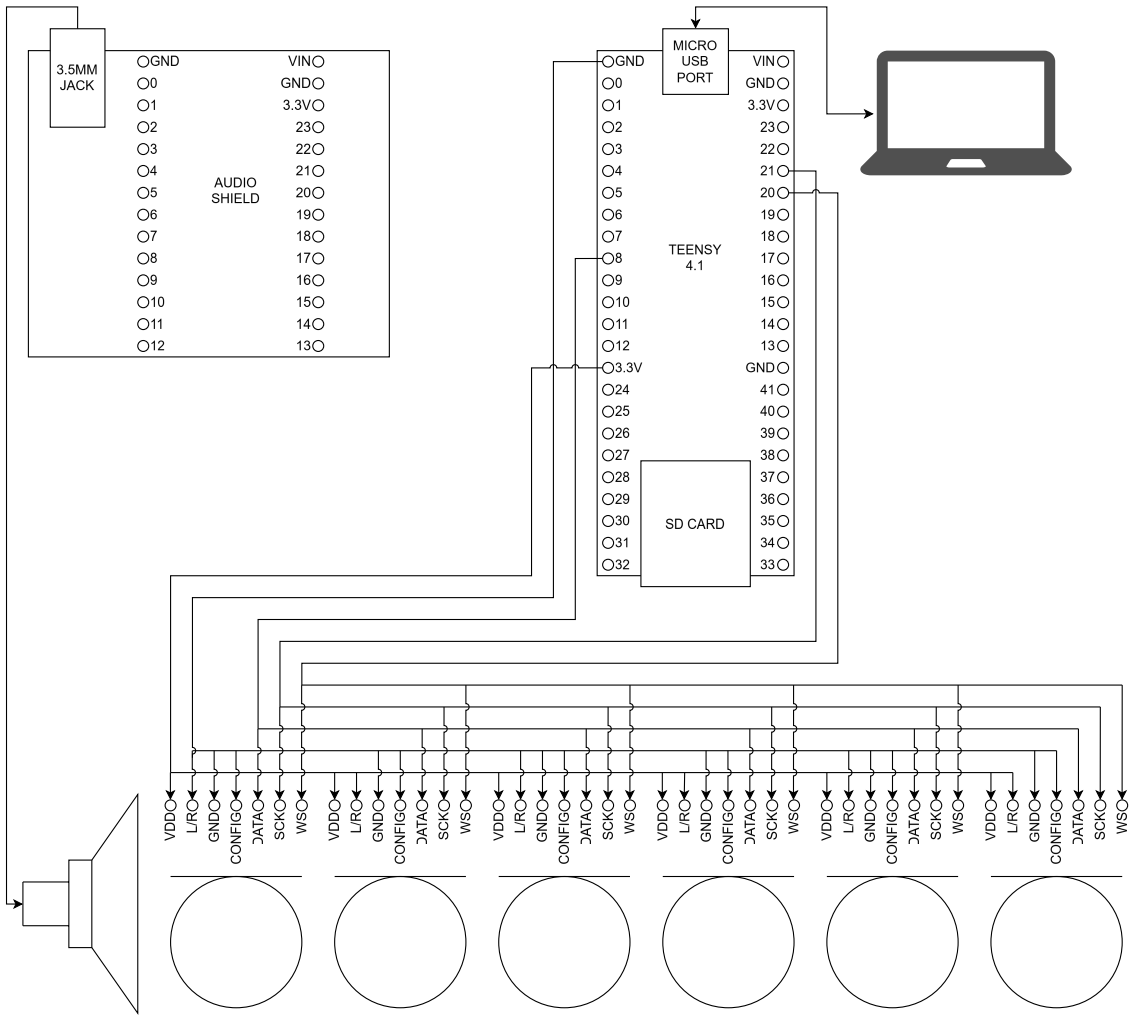


Figure 5.31: Test setup for measuring using 6 microphones on the Teensy 4.1. Notice that all 6 microphones are attached to the same I2S line, but are paired in L/R.

Actual Testing Within the very first FFT test, it became apparent that something was off. Channel 1/2 consistently yielded satisfactory measurements, whereas channel 3/4 was sporadic, and channel 5/6 was silent all along. By reading up on the documentation for the microphone itself, it became apparent that they do not support TDM scheduling. By further investigation and several code refactors, it also became apparent that it wasn't possible to address any of the microphones specifically. The only supported multi-microphone communication on the same I2s channel was stereo with 2 microphones. In the datasheet for the microphone, an array mode is mentioned, but looking further into it, the "array" would only consist of stereo data, and does not seem to be capable of daisychaining several microphones together in any manner.

Because of the inherent issues with the lack of TDM scheduling, a sequential approach is attempted. The sequential approach is structured to counter any issues posed by the lack of TDM scheduling. The main issue is the presence of more than two datalines in, on the same I2S channel. By having more than 2 datalines in, the Teensy defaults to a non-reading state, making all measurements 0.0.

In an effort to ensure only 2 datalines were "on" at a time, the VDD cable from all 6 microphones was attached to digital pins pulled low, until the specific microphone set (channel 1/2, 3/4, or 5/6) should be turned on by a high signal. This would resemble a

slowed-down version of the TDM scheduling, as it practically made the Teensy switch between each microphone set on a schedule. Within the schedule, time buffers were placed to allow microphone initialization and power-off sequences (20ms each). Despite no power being sent to the offline microphones, the data lines were still live, falling back to the 0.0 measurements. Instead of setting the microphones low, tri-stating was also tried, setting the "off" microphones at a high impedance, effectively disconnecting them. Unfortunately, this also proved unsuccessful.

The next step taken to mitigate these issues was to rewrite the I2S connections within the Teensy documentation, allowing a virtual third I2S data line to be made. While some progress was made, the implementation was too inconsistent and, in general, too unstable to be a viable option, as it simultaneously corrupted data on the I2S2 bus.

As initial trials were unsuccessful, rubber-ducking the problem led to the realization that if the data lines themselves could be switched on/off independently of the microphones on/off state, the problem could hopefully be handled. To make the solution as elegant as possible, the HI201HS quad CMOS analog switch was implemented. Unfortunately, the switch could not function at 3.3V logic. This led to the implementation of a DG445 quad SPST switch. When tested with an LED, this analog switch showed promise and seemed to function as intended. However, nothing came through the switch when testing with the I2S signal. This pattern could be replicated on all 4 channels of the switch, being capable of powering an LED, but not transmitting I2S signals.

In a last effort to measure with 6 microphones on a single I2S line, mechanical relays were tried. Finally, a signal could be seen, a signal could be seen on all 6 datalines when implemented sequentially. While not being the most elegant solution, it is the only functional solution to this very specific problem, which in most other cases would have been solved by using microphones capable of TDM. The final code for FFT measurements can be found on Github here, and in the attached files as "XXXX".

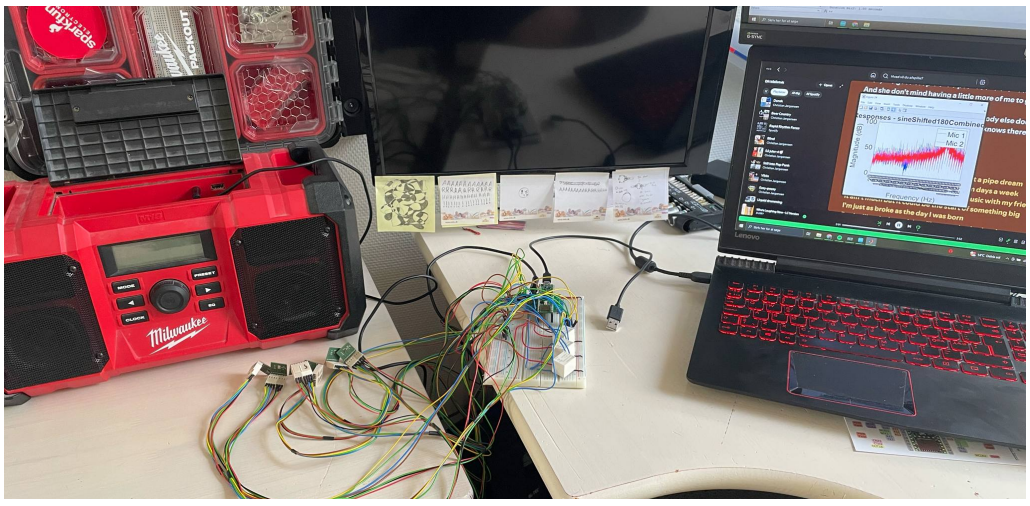


Figure 5.32: Real world setup for testing all 6 microphones sequentially using relays.

When it came to measuring impulse responses across all 6 microphones, the audio out signal had become quite noisy. The fault was found to be the I2S line chosen for audio in, as it shared the line with audio out. While this wasn't a problem in previous iterations, attaching 6 additional clocks to the I2S line yielded noisy output. The fault was avoided by changing the audio into I2S2, instead of I2S1. After this implementation, proper tests could be made. The final hardware schematic can be seen in figure 5.33,

and the real-world setup can be seen in figure 5.32. The code can be seen on GitHub here, or in the attached files as "xxxx".

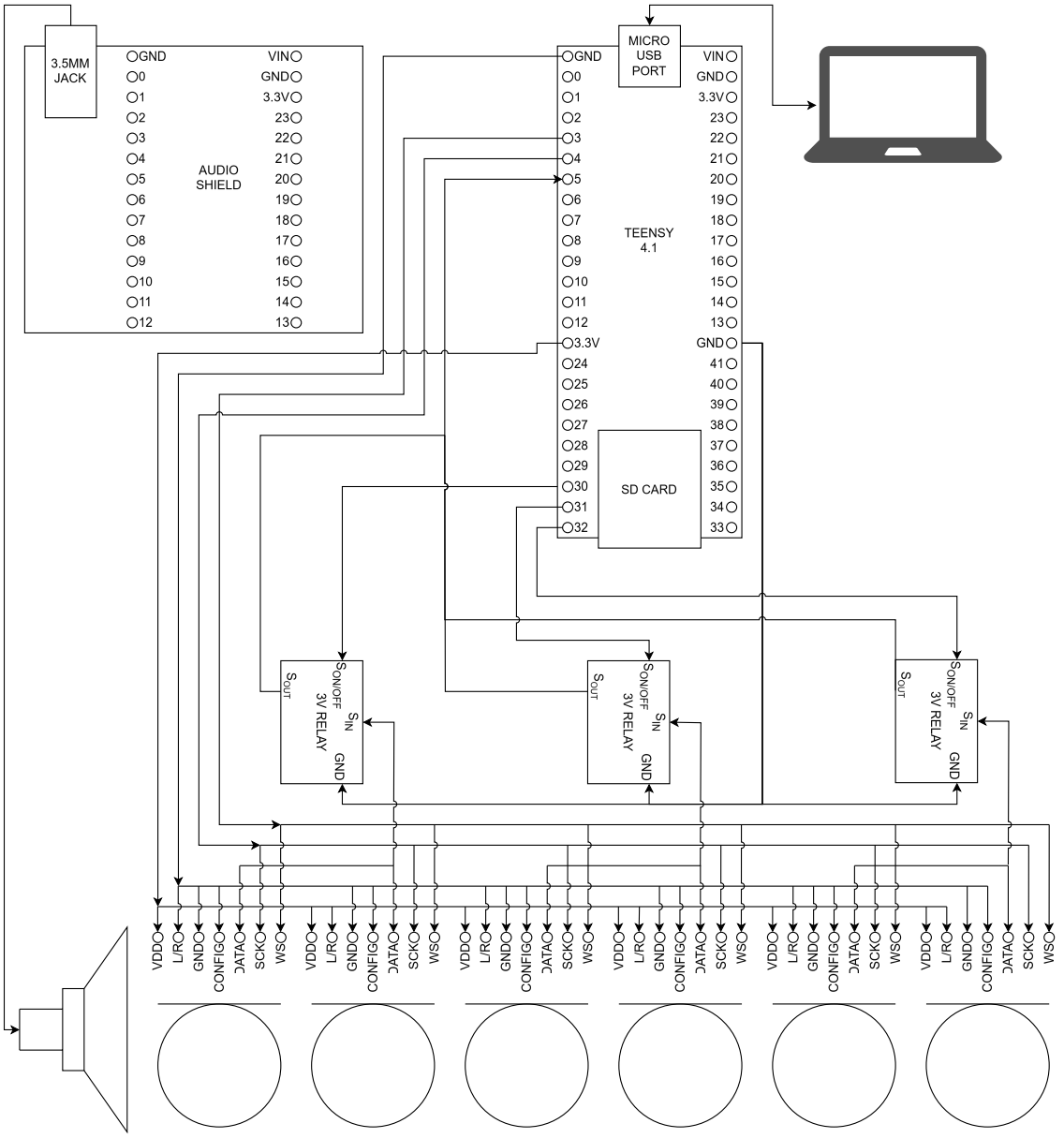


Figure 5.33: Schematic for measuring with 6 microphones sequentially using relays.

Test Results With software and hardware in place, the Teensy is capable of measuring across 6 microphones sequentially in pairs of 2 microphones. As in previous tests, serial output has been monitored. However, instead of showing a large section of the serial output, small snippets will show that each channel is usable. This can be seen in figure 5.34.

In figure 5.35, frequency domain responses of each channel subjected to puretone sinusoids, phase-shifted sines, sinesweeps, and white noise can be seen. From the plots, it can be deduced that the current implementation of measuring microphone pairs sequentially functions exactly as desired. While some differences are present between responses of each microphone set, it is considered natural, as the microphones are not placed 100% alike during this test, all the while the microphones still are not calibrated.

```
15:42:00.938 -> Channel 1:
15:42:00.938 -> 400.0-420.0: 0.000610  420.0-440.0: 0.000305  440.0-460.0: 0.000366  460.0-480.0: 0.000488  480.0-500.0: 0.000000
15:42:00.938 -> Channel 2:
15:42:00.938 -> 400.0-420.0: 0.000305  420.0-440.0: 0.000427  440.0-460.0: 0.000183  460.0-480.0: 0.000366  480.0-500.0: 0.000183
15:42:00.938 -> Channel 1:
15:42:00.938 -> 400.0-420.0: 0.000549  420.0-440.0: 0.000366  440.0-460.0: 0.000244  460.0-480.0: 0.000488  480.0-500.0: 0.000122
15:42:00.938 -> Channel 2:
15:42:00.938 -> 400.0-420.0: 0.000244  420.0-440.0: 0.000305  440.0-460.0: 0.000183  460.0-480.0: 0.000366  480.0-500.0: 0.000244
```

(a) Serial output of background noise measured by channel 1/2.

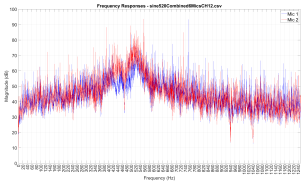
```
15:42:16.090 -> Channel 3:
15:42:16.090 -> 400.0-420.0: 0.000122  420.0-440.0: 0.000305  440.0-460.0: 0.000183  460.0-480.0: 0.000183  480.0-500.0: 0.000122  500.0-520.0: 0.000183
15:42:16.124 -> Channel 4:
15:42:16.124 -> 400.0-420.0: 0.000427  420.0-440.0: 0.000244  440.0-460.0: 0.000366  460.0-480.0: 0.000366  480.0-500.0: 0.000122  500.0-520.0: 0.000305
15:42:16.124 -> Channel 3:
15:42:16.124 -> 400.0-420.0: 0.000183  420.0-440.0: 0.000244  440.0-460.0: 0.000061  460.0-480.0: 0.000244  480.0-500.0: 0.000122  500.0-520.0: 0.000122
15:42:16.124 -> Channel 4:
15:42:16.124 -> 400.0-420.0: 0.000427  420.0-440.0: 0.000244  440.0-460.0: 0.000244  460.0-480.0: 0.000366  480.0-500.0: 0.000061  500.0-520.0: 0.000366
```

(b) Serial output of background noise measured by channel 3/4.

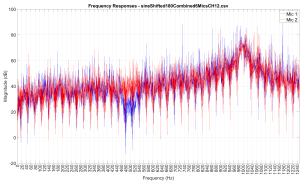
```
15:42:36.512 -> Channel 5:
15:42:36.512 -> 400.0-420.0: 0.000183  420.0-440.0: 0.000122  440.0-460.0: 0.000244  460.0-480.0: 0.000122  480.0-500.0: 0.000122
15:42:36.512 -> Channel 6:
15:42:36.512 -> 400.0-420.0: 0.000244  420.0-440.0: 0.000366  440.0-460.0: 0.000122  460.0-480.0: 0.000305  480.0-500.0: 0.000183
15:42:36.512 -> Channel 5:
15:42:36.512 -> 400.0-420.0: 0.000244  420.0-440.0: 0.000000  440.0-460.0: 0.000244  460.0-480.0: 0.000122  480.0-500.0: 0.000122
15:42:36.512 -> Channel 6:
15:42:36.512 -> 400.0-420.0: 0.000244  420.0-440.0: 0.000305  440.0-460.0: 0.000183  460.0-480.0: 0.000366  480.0-500.0: 0.000122
```

(c) Serial output of background noise measured by channel 5/6.

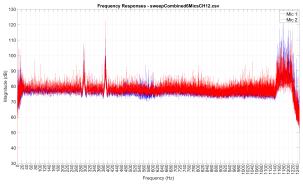
Figure 5.34



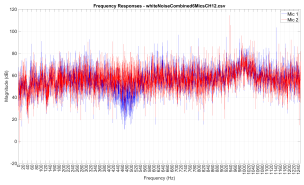
(a) Impulse response in the frequency domain of channel 1/2 at 520Hz.



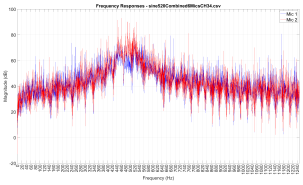
(b) Impulse response in the frequency domain of channel 1/2 at 1kHz phase shifted 180 degrees.



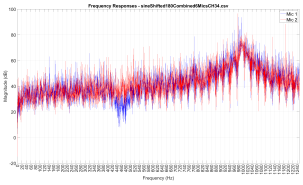
(c) Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz on channel 1/2.



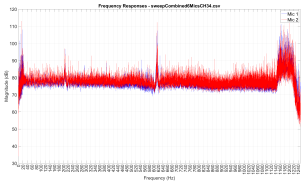
(d) Impulse response in the frequency domain of white noise on channel 1/2.



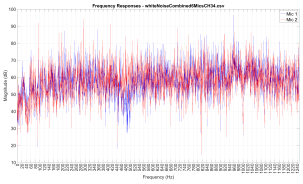
(e) Impulse response in the frequency domain of channel 3/4 at 520Hz.



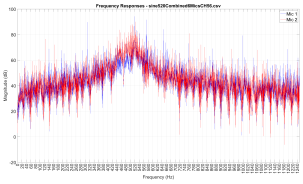
(f) Impulse response in the frequency domain of channel 3/4 at 1kHz phase shifted 180 degrees.



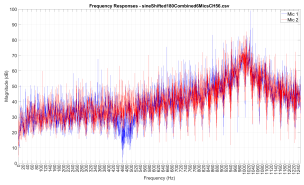
(g) Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz on channel 3/4.



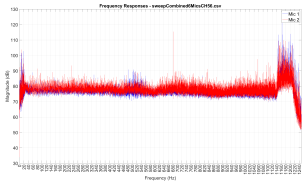
(h) Impulse response in the frequency domain of white noise on channel 3/4.



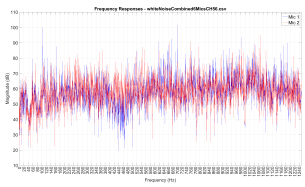
(i) Impulse response in the frequency domain of channel 5/6 at 520Hz.



(j) Impulse response in the frequency domain of channel 5/6 at 1kHz phase shifted 180 degrees.



(k) Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz on channel 5/6.



(l) Impulse response in the frequency domain of white noise on channel 5/6.

Figure 5.35

Summary

Despite many implementation issues spanning hardware and software, a final version is now capable of measuring across all 6 microphones and saving impulse responses to .CSV format. As shown by the plots in figure 5.35, the system is capable of measuring broadband signals and creating real-time FFTs within the frequency band 20Hz to 1220Hz. The final implementation can meet all specifications made in the beginning of this section, and can therefore be deemed functional to full extend.

5.4 Establishing an Algorithm For Data Handling

With the Teensy capable of creating and measuring sound while logging data, the next step is implementing an algorithm capable of calculating the impulse responses and transfer functions, making MATLAB obsolete. The data handling will comprise 2 parts:

- Impulse response and transfer function calculation
- Applying the transfer matrix method

5.4.1 Impulse Response and Transfer Function Calculating on Teensy 4.1

Calculating the impulse response and transfer functions in general is detrimental to calculating the transfer matrix method later on. As the desire is to create an all-in-one solution, it is desired to port this functionality to the Teensy, completely removing the need for a PC to compute anything. While the functionality introduced in section 5.3 makes it possible to derive all the necessary data needed for all further data handling in MATLAB or Python, it would be more user-friendly if the same functionality were implemented directly on the Teensy. A MATLAB script capable of doing all of the desired signal processing can be found here or in the attached files as "XXXX".

Specification

To facilitate further data processing, the Teensy program must be able to:

- Calculate impulse responses and transfer functions[2] from a signal 1 second in length
- Calculate impulse responses and transfer functions from a signal 10 seconds in length
- Calculate impulse responses and transfer functions from a 16-bit MLS (131071 samples)
- Calculate impulse responses and transfer functions from a 19-bit MLS (5248765 samples)
- Interpret data from .CSV format
- Save data in .CSV format

[2]Included in the transfer function should be: x, y1, y2, h1, h2, X, Y1, Y2, H1, H2, H1(dB), H2(dB), f, and phase in radians.

Design

Apart from the above-mentioned, the program has a major hurdle that needs to be overcome: memory restrictions on the Teensy. The Teensy only has 1 MB of RAM available, split into two chips of 512 kB each. This leaves room for 131.072 32-bit floats, which is sufficient for small sound samples, but by no means can accommodate the large memory needs of several seconds of data at 44.1 kHz sampling. Considering that the rest of the program also has to fit within this 1 MB, the capabilities are restricted even further.

To mitigate this issue, several solutions have been tested in an effort to find a viable solution. The first idea was implementing an STFT, sweeping through the signal. While an on-the-surface functioning STFT was implemented, it did not yield the correct results compared with MATLAB, no matter how much it was tweaked. The next iteration was to use the overlap-save method, but memory constraints led to stack overflows and other memory-related errors. The overlap-add method was tried as well, but yielded similar results. Furthermore, neither the overlap-add nor the overlap-save method yielded the correct results, compared to IR and FFTs computed by MATLAB. To utilize the installed SD card and all 16GB available, an out-of-core FFT approach was tried as well. Unfortunately, due to the added write/read times between SD and the teensy rather than the fast RAM communication, abhorrently long computation times were the only result. The fastest running version took more than 6 hours to compute the FFT and IR for a signal 1 second in length. However, most of the data was correct or at least within an acceptable deviation.

Up until now, the "arm_cfft_radix4_f32" framework has been used for FFT computation, and a fully functional version analysing 2048 samples has been made, with results matching MATLAB's exactly. As 2048 samples are far from the desired signal lengths, "kissFFT" is used instead. While kissFFT is far slower than the arm implementation, it has a significant benefit: with some small modifications in the main header file, it can be forced to store all buffers in PSRAM rather than RAM. Therefore, 8MB PSRAM is soldered onto the Teensy, and by doing so, the Teensy can now analyze signals containing 32768 rather than 2048 samples.

While this is still far from the desired signal length, it is an improvement. The current issue limiting the signal length is that the actual FFT computation still happens in the built-in RAM, and efforts to move this computation to PSRAM, resulted in multiple errors. Therefore, the version capable of analyzing 32768 samples is used. While this version does not match MATLAB's result exactly, it first becomes erroneous at the 6th decimal, which is acceptable, as it most likely stems from the Teensy sketch truncating values to 6 decimals.

Implementation

The FFT, IFFT, and resulting impulse response extraction is implemented through a customized version of the Kiss_FFT function and in general a modified version of the entire library[3]. The impulse response calculation is placed within a standalone function, for easier implementation in future code-ensembles, however, helper functions should also be remembered along with the modified KissFFT libraries.

[3]To perform tests, it is important to use the modified version found here, as the regular KissFFT library will not allow analysis of 32768 samples.

All buffers are placed in PSRAM using the "kiss_fft_alloc_psram()" call, allowing the Teensy to calculate FFTs using the PSRAM rather than RAM. To ensure compatibility with FFT demands, all signals are zero-padded to the next power of two, yielding greater precision as a bonus.

The transfer function is calculated for each frequency as a complex division of the output spectrum by the input spectrum with a small epsilon to avoid dividing by zero. The results are again stored in a buffer placed in PSRAM.

The inverse FFT is then computed, yielding the time domain impulse response. With all data present, it can be written into the .CSV file, allowing future calculations to be made. The format in the .CSV is: "FreqHz, Xreal, Ximag, Y1real, Y1imag, H1real, H1imag, H1magdB, H1phaserad, h1Impulseresponse, Y2real, Y2imag, H2real, H2imag, H2magdB, H2phaserad, h2Impulseresponse".

As the entirety of the program is more than 250 lines, it is easier to follow, reading it all together in GitHub here or in the attached files as "XXXX".

Test

Testing the FFT algorithm will be done by performing the FFT, IFFT, and impulse response calculations on both the Teensy and in MATLAB, before comparing results. As has already been mentioned in 5.4.1/5.4.1, several configurations have been tried and tested before the final iteration was chosen. This also means that the code has been implicitly tested prior to this documented test.

Test Setup The test setup is rather simple, as the Teensy only needs an SD card with a chosen file for analysis, and MATLAB needs access to the same file. The Teensy will run "IRCalculatorTeensyKissFFT3Columns.ino", which can be found here, and MATLAB will be running "impulseResponseCalculatorAllSignals6Mics.m", which can be found here. Both programs can also be found in the attached files. The file chosen for comparative analysis is "sweepEggShellFoamCH12.csv", which will yield the FFT and impulse response of a sweep from 20Hz to 1220Hz over 0.743 seconds (fitting 32768 samples at 44.1kHz).

Actual Testing As all actual testing has been performed and described in 5.4.1, nothing will be noted here.

Test Results In table 5.5 and 5.6, the first 8 frequency bins can be seen for the Teensy-derived transfer functions and impulse responses. In table 5.7 and 5.8, the same 8 first frequency bins for the MATLAB-derived transfer functions and impulse responses can be seen. By comparing these tables, it is easily seen that the Teensy implementation carries the same numerical precision as the MATLAB version.

FreqHz	H1real	H1imag	H1magdB	h1impulse
0.000000	0.008739	0.000000	-41.170422	-0.001848
0.672913	-0.025799	0.000986	-31.761759	-0.006477
1.345825	0.021090	0.013610	-32.006523	0.002999
2.018738	-0.017706	-0.031557	-28.829437	0.000808
2.691650	-0.015686	0.028916	-29.656921	-0.006284
3.364563	0.021312	-0.016109	-31.464848	0.005394
4.037476	-0.026087	-0.014995	-30.431520	-0.002170

Table 5.5: Frequency and transfer function components for H1 calculated by the Teensy.

FreqHz	H2real	H2imag	H2magdB	h2impulse
0.000000	0.005742	0.000000	-44.819229	-0.001785
0.672913	-0.024608	0.001991	-32.150085	-0.006137
1.345825	0.019005	0.013101	-32.734226	0.002833
2.018738	-0.016692	-0.029105	-29.485764	0.000658
2.691650	-0.015721	0.027667	-29.945477	-0.005800
3.364563	0.021102	-0.013700	-31.986082	0.005067
4.037476	-0.023394	-0.015536	-31.031210	-0.002034

Table 5.6: Frequency and transfer function components for H2 calculated by the Teensy.

FreqHz	H1real	H1imag	H1magdB	h1impulse
0.000000	0.008739	0.000000	-41.170422	-0.001848
0.672913	-0.025799	0.000986	-31.761759	-0.006477
1.345825	0.021090	0.013610	-32.006523	0.002999
2.018738	-0.017706	-0.031557	-28.829437	0.000808
2.691650	-0.015686	0.028916	-29.656921	-0.006284
3.364563	0.021312	-0.016109	-31.464848	0.005394
4.037476	-0.026087	-0.014995	-30.431520	-0.002170

Table 5.7: Frequency and transfer function components for H1 calculated by MATLAB.

FreqHz	H2real	H2imag	H2magdB	h2impulse
0.000000	0.005742	0.000000	-44.819229	-0.001785
0.672913	-0.024608	0.001991	-32.150085	-0.006137
1.345825	0.019005	0.013101	-32.734226	0.002833
2.018738	-0.016692	-0.029105	-29.485764	0.000658
2.691650	-0.015721	0.027667	-29.945477	-0.005800
3.364563	0.021102	-0.013700	-31.986082	0.005067
4.037476	-0.023394	-0.015536	-31.031210	-0.002034

Table 5.8: Frequency and transfer function components for H2 calculated by MATLAB.

Summary

As shown in tables 5.5-5.8, the Teensy-derived impulse responses are every bit as good as the MATLAB implementation. Unfortunately, as only 32768 samples can be processed, it can already be determined that only two of the specifications made in 5.4.1 can be met.

5.4.2 Applying the Transfer Matrix Method

To get any sort of meaningful data from all of the previous processes, the transfer matrix method has to be applied to the signals. As the microphones are not intended to be switched around in this particular impedance tube, the frequency domain deconvolution or impulse-response-based estimates will be used, as per section 8.6.3 and 8.6.4 in DS ISO 10534-2 2023. These approaches allow the use of the already calculated transfer functions, FFTs, and impulse responses found in section "Impulse Response and Transfer Function Calculating on Teensy 4.1". By doing so, the transfer matrix method should be applicable directly to the Teensy.

Specification

To conform to the standard while meeting the demands of this project, the transfer matrix method must follow these specifications:

- Take x_1 as input
- Take physical sizes of the tube as input
- Calculate k_0
- Calculate H_{12} from formula 3.52
- Calculate H_C from formula 3.45
- Calculate H_I
- Calculate H_R
- Calculate the reflection coefficient for a given frequency
- Calculate the absorption coefficient for a given frequency
- Calculate the impedance of a material at a given frequency
- Calculate the admittance of a material at a given frequency
- Save all data in .CSV format

Design

To ensure a functional transfer matrix method is applied to the Teensy, a matlab script is made prior to the Teensy sketch. The initial matlab script can be found here on GitHub. It is structured to import data from the FFT and IR .CSV made by the Teensy, then calculate the wave number k_0, before the transfer function H_{12} is found along with H_R and H_I. With those variables in place, the reflection coefficient can be found, and later the absorption coefficient. Having all variables available, the impedance and admittance can even be found. Lastly, all variables and acoustic properties are printed into a .CSV file for the frequency bucket nearest 1/3 octave frequencies.

5.4.3 Something Is Wrong

While the above design should be straightforward plug and play, it isn't. The fault is believed to be that the microphones or associated system components do not exhibit identical phase responses. This is the conclusion after having tried more iterations than is worth counting of various MMM matlab scripts on a host of different signals. The consistent errors are:

- Reflection coefficient around 1 no matter the material, signal type, or MMM method used.
- Absorption coefficient less than 0, almost no matter the material, signal type, or MMM method used.
- Even after aligning phases puretone sinusoids at one frequency, all other frequencies would be wildly out of phase.
- Close to realistic results could in some cases be made, if the real part of $|r|$ was used, instead of the absolute value of r. This does not follow the standard and is therefore not applicable.
- H_I and H_R could be correctly calculated, but $|r|$ is still wrong, most likely caused by erratic H_{12} values

- Despite having tried windowing methods, zero-padding, and calibration routines given by the DS ISO 10534-2 2023, H_{12} continues to be all over the place, with values spanning +-170, and not 0.5-1.5 as expected.

In the following, the diagnostic efforts made to locate and handle all of the errors are described in depth:

Despite rigorous compliance with the measurement standard and high mechanical precision in the physical setup, the anomalies persist. The following analysis sums up the key observations, suspected causes, attempted solutions, and current conclusions.

Observed Issues

As already mentioned, the most problematic issue is the magnitude of the reflection coefficient $|r|$ consistently lies around 1, resulting in alpha values near or below 0. Most problematic is it, when $|r|$ is more than 1, as this implies the tube actively increasing signal amplitude, despite having dampening materials inserted. The calculated transfer function $H_{12} = H_{s2}/H_{s1}$ exhibits highly erratic behavior in some frequency bands, occasionally showing magnitudes far exceeding 1, sometimes even +-100. This occurs despite the theoretical wave coefficients H_I and H_R being well-behaved and stable in most iterations of the MMM scripts. Somewhat realistic results could be obtained if only the real part of $|r|$ was used, but this was only applicable to specific frequency bands, and not the same frequency bands across varying materials. Furthermore, this approach does not follow the methodology set by the DS/ISO 10534-2 standard, and is therefore not applicable.

Verified Physical Setup

The physical setup has been verified to meet all expected tolerances. Microphone positions have been measured and verified to be within their nominal values. The impedance tube itself has been constructed to meet the tolerances of DS/ISO 10534-2, with the weight of the walls being the only exception. Despite the weight not being up to standard, the impedance tube sounds more dead than the standing wave tube supplied with Brüel and Kjærs Type 4002 standing wave apparatus. The amplification electronics were validated by monitoring speaker drive signals across measurements, showing a maximum variation of 0.5mV. Additionally, the microphone cables are matched within 5cm in length and are routed together to minimize differential noise. In terms of experimental procedure, all measurements have been conducted without switching microphones, which is allowed by the standard when using the frequency deconvolution or impulse response estimate. Even so, a separate verification using the standing wave method gave absorption curves that matched expectations, suggesting the tube behaves reasonably under different analysis methods. This is further described in section 6.

Suspected Causes and Analysis

Initial suspicions were a typo in the matlab script or a simple hardware issue, such as switching channels between mic1 and mic2. However, no typos could be found, and it made no difference if the microphone channels were switched around.

SNR was scrutinized next, as signals measured were a factor 10 smaller than what what inputted to the speaker. Nevertheless, measured signals are more than a factor 10 higher than background noise. This can be seen in figure **??**, where white noise and

background noise are both displayed. Furthermore, the Brüel & Kjær hardware connected to the speaker also shows an increase of far more than 30dB when any signal is played. Therefore, the SNR is deemed to be compliant with the DS/ISO standard specifying a ratio of 10 dB.

As mentioned at the beginning of this section, phase alignment errors were also found to be a major issue. Several alignment methods were tried, including:

- Phase slope estimation via linear regression on unwrapped phase.
- Cross-correlation in the time domain.
- Fractional delay filtering in the frequency domain.

However, even with precise estimation of sample delay (e.g., 176 samples), the resulting FFT phase difference could be matched for some frequencies, such as 1kHz, while others would be far off.

Troubleshooting and Diagnostic Results

A large number of diagnostic scripts were implemented to analyze impulse responses, FFT magnitudes, transfer functions, alignment residuals, and phase compositions. Despite impulse responses being nearly identical when plotted (largest difference being a slight offset, corresponding with differences in microphone placement), FFTs of said impulse responses often diverged by up to more than 20 dB for some frequency bands.

Moreover, while the time domain phase alignment seemed to work superficially, allowing impulse responses to be visually overlaid, the frequency domain phase was still inconsistent. This points in the direction of phase response mismatches hiding within the hardware or otherwise uncorrected group delay differences.

Summary

In conclusion, the MMM is incredibly sensitive to hardware and software irregularities, even more than anticipated when designing the system. The current issues arise not from an individually identifiable issue, but are suspected to be mainly caused by wrongful phase responses, leading to reflection and absorption coefficients that do not correlate to measurements made with the standing wave method, while also being physically impossible.

Therefore, the transfer matrix method cannot be implemented, and will therefore not be tested or evaluated further, until the phase difference can be explained.

5.5 Integration and the Lack Thereof

As all Teensy code snippets have been used across each other in various formats to perform previous tests, a designated integration process is left out. This is deemed viable as all functions are designed to be implementable in other programs, as long as the correct libraries and variables are implemented as well. Therefore, a single program is created, merging all functionality from previous programs. This program is named "combinedAcceptanceTest.ino", and can be found on GitHub here or in the attached files. As the name implies, testing will be described in the acceptance test, starting page

102. As mentioned in 5.4.2, the transfer matrix method is not applied and will therefore not be integrated into the final acceptance test.

6 Acceptance test

General acceptance test

Sammenligningstest med Bruel og Kjær 4002 Standbølge Apparat af forskellige materialer.

7 Discussion

7.1 The Transfer Matrix Method

misforståelser udfordrende at implementere selv i MATLAB Blev ikke implementeret på Teensy pga tidsproblemer

7.2 Porting Complex Math to a Cheap MCU

memory troede den var klar til fft men nej

7.3 Physical Hurdles

akryl mikrofoner i2s kanaler

7.4 The Desire to Understand Everything

7.5 Timetable

7.6 Working Alone

8 Conclusion

"?"

Bibliography

[1] Francesco Aletta and Jian Kang. "Historical Acoustics Relationships between People and Sound over Time". In: (). URL: www.mdpi.com/journal/acoustics.

[2] *Akustiske kunstværker til alle hjemmets rum | Arturel*. URL: https://arturel.dk/.

[3] *Akustikbilleder | Få ro på kontoret & i hjemmet*. URL: https://www.dansklydisolering.dk/akustik/akustikbilleder.

[4] F. Alton Everest and Ken C Pohlmann. *Master Handbook of Acoustics, Seventh Edition*. eng. Seventh edition. New York, N.Y: McGraw-Hill Education, 2022. ISBN: 9781260473599.

[5] Gerhard. Müller and Michael. Möser. *Handbook of Engineering Acoustics*. eng. Ed. by Gerhard. Müller, Michael. Möser, and SpringerLink (Online service). Elektronisk udgave. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. ISBN: 9783540694601.

[6] Douglas. Self. *Audio engineering : know it all.* eng. Ed. by Richard. Brice et al. Boston: Elsevier, 2009. ISBN: 9781856175265.

[7] Heinrich Kuttruff. *Room Acoustics*. 3rd ed. 1991.

[8] Heinrich Kuttruff and Michael Vorländer. *Room Acoustics*. eng. 7th edition. Boca Raton: CRC Press, 2025. ISBN: 1032486104. DOI: 10.1201/9781003389873.

[9] "Akustik-Måling af rumakustiske parametre-Del 1: Rum til optraeden Acoustics-Measurement of room acoustic parameters-Part 1: Performance spaces". In: (2009).

[10] *4 - Room Acoustics | Course Chapters | Applications in Communication Acoustics | edX*. URL: https://learning.edx.org/course/course-v1:RWTHTUMx+CA101.2x+3T2024/block-v1:RWTHTUMx+CA101.2x+3T2024+type@sequential+block@98c70b51aef741e8baa200b9b3cfd75d/block-v1:RWTHTUMx+CA101.2x+3T2024+type@vertical+block@49f5c48c68a2417faa5ac93bc23dad48.

[11] *5 - Room Simulation | Course Chapters | Applications in Communication Acoustics | edX*. URL: https://learning.edx.org/course/course-v1:RWTHTUMx+CA101.2x+3T2024/block-v1:RWTHTUMx+CA101.2x+3T2024+type@sequential+block@3807b6e727d34ac1b330fb277796fccf/block-v1:RWTHTUMx+CA101.2x+3T2024+type@vertical+block@023ac225abd04c808862d53129925533.

[12] *8 - Psychoacoustics in Product Development | Course Chapters | Applications in Communication Acoustics | edX*. URL: https://learning.edx.org/course/course-v1:RWTHTUMx+CA101.2x+3T2024/block-v1:RWTHTUMx+CA101.2x+3T2024+type@sequential+block@d5ffb03395564336b1d50adfe21cb2ca/block-v1:RWTHTUMx+CA101.2x+3T2024+type@vertical+block@3cc6f102482341fa91df33e762fad22b.

[13] "Akustik-Bestemmelse af lydabsorptionskoefficient og impedans i impedansrør-Del 1: Metode baseret på måling af standbølgeforhold Acoustics-Determination of sound absorption coefficient and impedance in impedance tubes-Part 1: Method using standing wave ratio". In: (2001).

[14] "DANSK STANDARD Akustik-Bestemmelse af akustiske egenskaber i impedansrør-Del 2: Lydabsorptionskoefficient og overfladeimpedans ved normalt lydindfald udført med to mikrofoner Acoustics-Determination of acoustic properties in impedance tubes-Part 2: Two-microphone technique for normal sound absorption coefficient and normal surface impedance". In: (2023). URL: www.ds.dk.

[15] M. David. Egan. *Architectural acoustics.* eng. J. Ross Publishing Classics. J. Ross Publishing, 2007. ISBN: 9781932159783.

[16] Christopher N. Brooks. *Architectural acoustics*. eng. Ed. by Christopher N. Brooks. Jefferson, N.C: McFarland & Company, 2003. ISBN: 9780786413980.

[17] Nikolaos M. Papadakis. *Advances in Architectural Acoustics*. eng. Ed. by Nikolaos M. Papadakis, Massimo Garai, and Stavroulakis Georgios. Basel: MDPI - Multidisciplinary Digital Publishing Institute, 2022. ISBN: 3-0365-4295-7.

[18] Jerry H Ginsberg. *Acoustics-A Textbook for Engineers and Physicists: Volume I: Fundamentals*. eng. 1st ed. Cham: Springer International Publishing AG, 2017. ISBN: 3319568434. DOI: 10.1007/978-3-319-56844-7.

[19] Fridolin P Mechel et al. *Formulas of Acoustics*. eng. 2nd Edition. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2008. ISBN: 9783540768326. DOI: 10.1007/978-3-540-76833-3.

[20] Robert D Blevins. *Formulas for Dynamics, Acoustics and Vibration*. eng. 1st ed. Wiley series in acoustics noise and vibration. Newark: John Wiley & Sons, 2016. ISBN: 1119038111. DOI: 10.1002/9781119038122.

[21] "Engineering and Intuition Serving the Soul of Music". In: ().

[22] *Room acoustic descriptors - RT, C50 and Strength/ Gain - Acoustic Bulletin*. URL: https://www.acousticbulletin.com/room-acoustic-descriptors-rt-c50-and-gain/.

[23] *Room acoustic parameters | Architectural Acoustics Class Notes | Fiveable*. URL: https://library.fiveable.me/architectural-acoustics/unit-2/room-acoustic-parameters/study-guide/8vMXLLFgm6jqnASi.

[24] "DANSK STANDARD Akustik-Normalkurver for toner med samme hørestyrkeniveau Acoustics-Normal equal-loudness-level contours". In: (2023). URL: www.ds.dk.

[25] *DS/EN ISO 389-7:2019 : Standard Distribute*. URL: https://sd.ds.dk/Viewer?ProjectNr=M335052&Status=60.60&Inline=true&Page=1&VariantID=.

[26] "Akustik-Fremgangsmåde til beregning af hørestyrke-Del 1: Zwickers metode Acoustics-Methods for calculating loudness-Part 1: Zwicker method". In: (2017).

[27] *sone2phon*. URL: https://se.mathworks.com/help/audio/ref/sone2phon.html.

[28] *3 - The Human Auditory System | Course Chapters | Fundamentals of Communication Acoustics | edX*. URL: https://learning.edx.org/course/course-v1:RWTHTUMx+CA101.1x+3T2024/block-v1:RWTHTUMx+CA101.1x+3T2024+type@sequential+block@b2d0223a20644dda975d4ff6bb1cfb84/block-v1:RWTHTUMx+CA101.1x+3T2024+type@vertical+block@6fb78eab34a74381a0a4a0617dc7277f.

[29] *2 - Binaural Technology | Course Chapters | Applications in Communication Acoustics | edX*. URL: https://learning.edx.org/course/course-v1:RWTHTUMx+CA101.2x+3T2024/block-v1:RWTHTUMx+CA101.2x+3T2024+type@sequential+block@16e85d086ada44b39b25de65da06e143/block-v1:RWTHTUMx+CA101.2x+3T2024+type@vertical+block@797de25693574764a44c59b010cbc147.

[30] *Sound Absorbing Materials : Zwikker,c : Free Download, Borrow, and Streaming : Internet Archive*. URL: https://archive.org/details/in.ernet.dli.2015.140695/page/n31/mode/2up.

[31] "Architectural Acoustics Briiel & Kjaer". In: ().

[32] Per V. Brüel. "Brüel & Kjær Technical Review 1955 - Standing wave apparatus type 4002". In: 1 (1955), pp. 3–24. URL: https://pearl-hifi.com/06_Lit_Archive/15_Mfrs_Publications/10_Bruel_Kjaer/02_Product_Data/4002_Standing_Wave_Apparatus.pdf.

[33] Per V. Bruël. "Brüel & Kjær Instructions and Applications 1955 - Standing wave apparatus type 4002". In: (Apr. 1955), pp. 25–47. URL: https://pearl-hifi.com/06_Lit_Archive/15_Mfrs_Publications/10_Bruel_Kjaer/02_Product_Data/4002_Standing_Wave_Apparatus.pdf.

[34] Per V. Brüel. "Brüel & Kjær Instruction Manual - Standing Wave Apparatus Type 4002". In: (Mar. 1979), pp. 48–72. URL: https://pearl-hifi.com/06_Lit_Archive/15_Mfrs_Publications/10_Bruel_Kjaer/02_Product_Data/4002_Standing_Wave_Apparatus.pdf.

[35] Per V. Brüel. *Technical Review 97-01*. 1997. URL: http://www.bruel-ac.com/tr/tr9701/TR9701.html.

[36] "Akustik-Måling af rumakustiske parametre-Del 2: Efterklangstid i almindelige rum Acoustics-Measurement of room acoustic parameters-Part 2: Reverberation time in ordinary rooms". In: (2008).

[37] *RT60 Reverberation Time - Svantek Academy*. URL: https://svantek.com/academy/rt60-reverberation-time/.

[38] *Reverberation Time*. URL: https://www.nti-audio.com/en/applications/room-building-acoustics/reverberation-time.

[39] *What is early decay time EDT? – Oceanus gas detection system include of the Fixed gas detector, Portable gas detector, Air quality monitor system*. URL: https://www.ocgasdetector.com/what-is-early-decay-time-edt/.

[40] "Akustik-Måling af lydabsorption i efterklangsrum Acoustics-Measurement of sound absorption in a reverberation room". In: (2003).

[41] *Sound and Vibration Handbook | Brüel & Kjær*. URL: https://www.bksv.com/downloads/svpocketehandbook/index.html.

[42] *Application of Transfer Matrix Method in Acoustics*. URL: https://www.comsol.com/paper/application-of-transfer-matrix-method-in-acoustics-8420.

[43] *DS/ISO 9613-1:1993 : Standard Distribute*. URL: https://sd.ds.dk/Viewer?ProjectNr=14249&Status=90.20&Inline=true&Page=1&VariantID=.

[44] *Acoustic measurements : Beranek, Leo Leroy : Free Download, Borrow, and Streaming : Internet Archive*. URL: https://archive.org/details/acousticmeasurem00inbera/page/890/mode/2up.

[45] "Standard Test Method for Measurement of Normal Incidence Sound Transmission of Acoustical Materials Based on the Transfer Matrix Method 1". In: 2611-09 (2009). DOI: 10.1520/E2611-09. URL: https://tajhizkala.ir/doc/ASTM/E2611-09.pdf.

[46] *E2611 Standard Test Method for Normal Incidence Determination of Porous Material Acoustical Properties Based on the Transfer Matrix Method*. URL: https://store.astm.org/e2611-24.html.

[47] "TMS320C64x DSP Library Programmer's Reference". In: (2003). URL: www.ti.com/automotive.

[48] *Linear-feedback shift register - Wikipedia*. URL: https://en.wikipedia.org/wiki/Linear-feedback_shift_register.

[49] *Lecture 52: Maximum Length Sequence and its Propertes - YouTube*. URL: https://www.youtube.com/watch?v=QG_WooY4nOA.

[50] *Maximum length sequence - Wikipedia*. URL: https://en.wikipedia.org/wiki/Maximum_length_sequence.

[51] *Random Numbers with LFSR (Linear Feedback Shift Register) - Computerphile - YouTube*. URL: https://www.youtube.com/watch?v=Ks1pw1X22y4.

[52] *LFSR: Linear Feedback Shift Register Basics for Pseudo Noise Generation - YouTube*. URL: https://www.youtube.com/watch?v=Antpjin7Hp4.

[53] Michael Vorländer and Malte Kob. "Practical aspects of MLS measurements in building acoustics". In: *Applied Acoustics* 52.3-4 (Nov. 1997), pp. 239–258. ISSN: 0003-682X. DOI: 10.1016/S0003-682X(97)00029-7.

[54] *hunecke.de | Maximum length diffusors*. URL: https://www.hunecke.de/en/knowledge/diffusors/maximum-length-diffusors.html.

[55] *mls*. URL: https://se.mathworks.com/help/audio/ref/mls.html.

[56] *chirp*. URL: https://se.mathworks.com/help/signal/ref/chirp.html.

[57] *chirp — SciPy v1.15.2 Manual*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.chirp.html?utm_source=chatgpt.com.

[58] *Chirp Signals*. URL: https://www.dspguide.com/ch11/6.htm.

[59] Resmi Suresh and Raghunathan Rengaswamy. "Time-frequency equivalence using chirp signals for frequency response analysis Time-frequency equivalence using chirp signals for frequency 1 response analysis". In: (2021). DOI: 10.21203/rs.3.rs-134038/v1.

[60] *Impedance Tube | Mecanum | ASTM E1050 ASTM E2611 ISO 10534-2*. URL: https://mecanum.com/measuring-instruments/conventional-tubes/.

[61] *Steel Weight Calculator*. URL: https://www.omnicalculator.com/construction/steel-weight.

[62] *Density (Unit weight) of Concrete - Civil Engineering*. URL: https://civiltoday.com/civil-engineering-materials/concrete/361-density-of-concrete.

[63] *Krydsfiner - Træ.dk*. URL: https://www.trae.dk/leksikon/krydsfiner/.

[64] *Poly(methyl methacrylate) - Wikipedia*. URL: https://en.wikipedia.org/wiki/Poly(methyl_methacrylate).

[65] *Glass Weight Calculator*. URL: https://www.omnicalculator.com/construction/glass-weight.

[66] *Digital skydelære, rustfri - Biltema.dk*. URL: https://www.biltema.dk/varktoj/male varktoj/skydelare/digital-skydelare-rustfri-2000040853.

[67] *Meterstok 59 | Hultafors*. URL: https://hultafors.com/da-dk/products/folding-rule-59?variant=55034856341890.

[68] *DMM-4026-B-I2S-R.pdf*. URL: https://docs.google.com/gview?url=https://api.puiaudio.com/filename/DMM-4026-B-I2S-R.pdf&embedded=true.

[69] *What are 1/1 and 1/3 Octave Bands and why are they used?* URL: https://www.castlegroup.co.uk/octave-bands/.

[70] "Center Frequencies and High/Low Frequency Limits for Octave Bands, 1/2-and 1/3-Octave Bands". In: ().

[71] *Octave Bands in Room Acoustics – Acoustic Fields*. URL: https://www.acousticfields.com/octave-bands-in-room-acoustics/.

[72] *Octave Band Frequencies*. URL: https://www.engineeringtoolbox.com/octave-bands-frequency-limits-d_1602.html.

Glossary

C50 Clarity 50 ms. 2, 22, 23

C80 Clarity 80 ms. 22

CSV Comma Separated Value. 49, 64, 66, 71, 73, 75, 78, 79, 82, 94, 96, 98

D50 Definition 50 ms. 2, 23

EDT Early Decay Time. 2, 21, 22

MLS Maximum Length Sequence. 40, 43–47, 50, 63–70, 72–74, 77

MMM Multi Microphone Method. 98–100

MSB Most significant byte. 43

RT60 Reverberation Time 60 dB. 2, 21, 22

SNR Signal to Noise Ratio. 46, 47, 77, 99, 100

SPL Sound pressure level. 4

T10 Reverberation Time 10 dB. 22

T20 Reverberation Time 20 dB. 21, 22

T30 Reverberation Time 30 dB. 21, 22

A Appendix

B | Mathematical Notations

B.1 Variables

$p(t)$ is the instantaneous sound pressure of the impulse response measured at a point in the room.
$p_{10}(t)$ is the instantaneous sound pressure of the impulse response measured at 10m in free field.
p_0 is $20\mu Pa$.
T_0 is 1 second.
L_{pE} is the sound pressure exposure level of $p(t)$.
$L_{pE,10}$ is the sound pressure exposure level of $p_{10}(t)$.
L_p is the sound pressure level averaged across every measurement point.
$L_{p,10}$ is the sound pressure level measured at 10 m in free field.
L_p is the sound pressure level averaged across every measurement point.
L_W is the sound power level of the sound source, and should be measured according to DS 3741.
$A = \sum S_i * \alpha_i$.
C_{t_e} is the early to late index.
t_e is the early time limit (e.g. 50 or 80 ms).
$p(t)$ is the measured sound pressure of the impulse response.
$c^2 = \kappa \frac{p_0}{\rho_0}$.
p is the sound pressure.
ρ_0 is the static gas density value ($\approx 1.2[\frac{kg}{m^3}]$ for air).
p_0 is the static sound pressure (measured in Pascal, where 1 $Pascal = 1\frac{kg}{m*s^2}$.
κ is the adiabatic component.
v is a vector representing particle velocity.
t is time.
ρ being the variable gas density.
$\hat{p}$ is amplitude.
k is the propagation constant $k = \frac{\omega}{c}$.
ω is the angular frequency, which can be used to obtain the temporal period $T = \frac{2*\pi}{\omega}$.
$f = \frac{\omega}{2*\pi} * \frac{1}{T}$ with the unit Hz.
$\rho_0 * c$ found by table values for ρ_0 and c in the correct medium.
θ and ϕ angles,
c^2 on both sides,
Q, being the rate in $\frac{m^3}{s}$ of "liquid" being expelled by the sound source located at $r = 0$. The dot above Q denotes differentation w.r.t. time (t).
r and frequency $\omega = k * c$.
$\frac{p}{v}$ tends asymptotically to $\rho_0 c$.
Γ_{nm} are the Fourier series coefficients of order n and degree m,

$h_n(kr)$ is the Hankel function of the first kind with order n,
$Y_n^m(\theta, \phi)$ are the spherical harmonics,
$\tilde{p}$ denotes root mean square of the sound pressure, i.e
$\sqrt{\bar{p^2}}$ and $\tilde{p_0} = 2 * 10^{-5}[\frac{N}{m^2}]$ is an internationally fixed value corresponding to the hearing threshold at 1000Hz,
N being loudness in sones and
L_N being loudness in phons
$x = 0$ and the wave will arrive from the negative direction,

B.2 Formulas

$$G = 10 * log10 \left(\frac{\int_0^\infty p^2(t)\, dt}{\int_0^\infty p_{10}^2(t)\, dt} \right) = L_{pE} - L_{pE,10}[dB] \tag{B.1}$$

$$L_{pE} = 10 * log10 \left[\frac{1}{T_0} * \int_0^\infty \frac{p^2(t)\, dt}{p_0^2} \right] [dB] \tag{B.2}$$

$$L_{pE,10} = 10 * log10 \left[\frac{1}{T_0} * \int_0^\infty \frac{p_{10}^2(t)\, dt}{p_0^2} \right] [dB] \tag{B.3}$$

$$L_{pE,10} = L_{pE,d} + 20 * log10(d/10)[dB] \tag{B.4}$$

$$L_p = L_{p,10}[dB] \tag{B.5}$$

$$G = L_p - L_W + 31dB \tag{B.6}$$

$$RT60 = \frac{0.161 * V}{S_i * \alpha_i} \leftrightarrow RT = \frac{0.161 * V}{A} \tag{B.7}$$

$$A = \sum S_i * \alpha_i \tag{B.8}$$

$$C50 = 10 * log10 \left(\frac{Energy(0 - 50ms)}{Energy(51ms - end)} \right) [dB] \tag{B.9}$$

$$C80 = 10 * log10 \left(\frac{Energy(0 - 80ms)}{Energy(81ms - end)} \right) [dB] \tag{B.10}$$

$$C_{t_e} = 10 * log10 \left(\frac{\int_0^{t_e} p^2(t)\, dt}{\int_{t_e}^\infty p^2(t)\, dt} \right) [dB] \tag{B.11}$$

$$C50 = 10 * log10 \left(\frac{D50}{1 - D50} \right) [dB] \tag{B.12}$$

$$D50 = \frac{\int_0^{0.050} p^2(t)\, dt}{\int_0^\infty p^2(t)\, dt} \tag{B.13}$$

$$c = 331.4 + 0.6 * T[\frac{m}{s}] \tag{B.14}$$

$$c^2 \Delta p = \frac{\partial^2 p}{\partial t^2} \tag{B.15}$$

$$c^2 = \kappa \frac{p_0}{\rho_0} \tag{B.16}$$

$$\Delta p = \frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2} \tag{B.17}$$

$$grad\ p = -\rho_0 * \frac{\partial v}{\partial t} \tag{B.18}$$

$$\frac{\partial p}{\partial x} = -\rho_0 \frac{\partial v_x}{\partial t} \tag{B.19}$$

$$\rho_0\ div\ v = -\frac{\partial \rho}{\partial t} \tag{B.20}$$

$$\frac{p}{p_0} = \kappa \frac{\rho}{\rho_0} = \frac{\kappa}{\kappa - 1} * \frac{\delta * T}{T + 273} \tag{B.21}$$

$$c^2 \frac{\partial^2 p}{\partial x^2} = \frac{\partial^2 p}{\partial t^2} \tag{B.22}$$

$$p(x,t) = F(c * t - x) + G(c * t + x) \tag{B.23}$$

$$p(x,t) = \hat{p} * e^{i*k*(c*t-x)} = \hat{p} * e^{i*(\omega*t-k*x)} \tag{B.24}$$

$$\lambda = \frac{2 * \pi}{k} \leftrightarrow \frac{2 * \pi}{\frac{\omega}{c}} \leftrightarrow \frac{2 * \pi * c}{\omega} \leftrightarrow \frac{c}{\frac{\omega}{2*\pi}} \leftrightarrow \frac{c}{f} \tag{B.25}$$

$$v(x,t) = \frac{1}{\rho_0 * c}[F(c * t - x) - G(c * t + x)] \tag{B.26}$$

$$\frac{p}{v} = \rho_0 * c \tag{B.27}$$

$$\rho_{0_{air}} * c_{air} = 1.2 \left[\frac{kg}{m^3}\right] * 343 \left[\frac{m}{s}\right] = 414 \left[\frac{kg}{m^2 s}\right] \tag{B.28}$$

$$I = \bar{p}\bar{v} = \frac{\bar{p^2}}{\rho_0 * c} \tag{B.29}$$

$$w = \frac{I}{c} = \frac{\bar{p^2}}{\rho_0 * c^2} \tag{B.30}$$

$$\Delta p = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 \frac{\partial p}{\partial r}\right) + \frac{1}{r^2 \sin\theta}\frac{\partial}{\partial \theta}\left(\sin\theta \frac{\partial p}{\partial \theta}\right) + \frac{1}{r^2 \sin^2\theta}\frac{\partial^2 p}{\partial \phi^2}. \tag{B.31}$$

$$\Delta p = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 \frac{\partial p}{\partial r}\right) \tag{B.32}$$

$$\Delta p = \frac{\partial^2 p}{\partial r^2} + \frac{2}{r}\frac{\partial p}{\partial r} \tag{B.33}$$

$$c^2\left(\frac{\partial^2 p}{\partial r^2}+\frac{2}{r}\frac{\partial p}{\partial r}\right)=\frac{\partial^2 p}{\partial t^2} \tag{B.34}$$

$$\frac{\partial^2 p}{\partial r^2}+\frac{2}{r}*\frac{\partial p}{\partial r}=\frac{1}{c^2}*\frac{\partial^2 p}{\partial t^2} \tag{B.35}$$

$$p(r,t)=\frac{\rho_0}{4*\pi*r}*\dot{Q}\left(t-\frac{r}{c}\right) \tag{B.36}$$

$$v_r=\frac{1}{4*\pi*r^2}\left[Q\left(t-\frac{r}{c}\right)+\frac{r}{c}*\dot{Q}\left(t-\frac{r}{c}\right)\right] \tag{B.37}$$

$$v_r=\frac{p}{\rho_0*c}*\left(1+\frac{1}{i*k*r}\right) \tag{B.38}$$

$$\frac{\partial^2 p}{\partial r^2}+\frac{2}{r}*\frac{\partial p}{\partial r}+\frac{1}{r^2}\left(\frac{1}{sin(\theta)}*\frac{\partial}{\partial*\theta}\left(sin(\theta)*\frac{\partial}{\partial*\theta}\right)+\frac{1}{sin^2(\theta)}*\frac{\partial^2}{\partial\phi^2}\right)=\frac{1}{c^2}*\frac{\partial^2 p}{\partial t^2} \tag{B.39}$$

$$p(r,\theta,\phi)=A*\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\Gamma_{nm}h_n(kr)Y_n^m(\theta,\phi) \tag{B.40}$$

$$Y_n^m(\theta,\phi)=\sqrt{\frac{(2n+1)}{4*\pi}*\frac{(n-|m|)!}{(n+|m|)!}}*P_n^{|m|}*(cos(\theta)*\begin{cases}cos|m|\phi & if\ \ m\geq 0\\ sin|m|\phi & if\ \ m<0\end{cases} \tag{B.41}$$

$$SPL=20*log10(\frac{\tilde{p}}{\tilde{p_0}}[dB] \tag{B.42}$$

$$N=\left(10^{\frac{L_N-40}{10}}\right)^{0.30103}\approx 2^{\frac{L_N-40}{10}} \tag{B.43}$$

$$L_N=40+log_2(N) \tag{B.44}$$

Phon	0	10	20	30	40	50	60	70	80	90	100
Sone	0 (Inaudible)	0.0625	0.125	0.25	1	2	4	8	16	32	64

Table B.1: Relationship between phons and sones.

- **Reflection Coefficient:**

$$R=\frac{I_r}{I_i} \tag{B.45}$$

where I_r is reflected intensity and I_i is incident intensity.

- **Diffusion Coefficient:**

$$D=1-\frac{\sum S(\theta)}{NS_{\text{ideal}}} \tag{B.46}$$

where $S(\theta)$ is the scattered energy in direction θ, and S_{ideal} is the ideally scattered energy.

- **Transmission Loss (TL):**

$$TL=10\log_{10}\left(\frac{I_i}{I_t}\right) \tag{B.47}$$

where I_t is the transmitted intensity.

Property	Definition	Typical Values
Reflection Coefficient (R)	Fraction of sound energy reflected	0.8 - 0.98 (Concrete), 0.1 - 0.5 (Foam)
Diffusion Coefficient (D)	Measure of even sound scattering	0.2 (Flat wall), 0.6 - 0.9 (Diffuser)
Transmission Loss (TL)	Reduction of sound through a barrier (dB)	25 dB (Drywall), 50+ dB (Concrete)
Resonance Frequency (f_r)	Frequency at which the wall vibrates	50 - 500 Hz (Depending on material)
Absorption Coefficient (α)	Fraction of energy absorbed	0.02 (Concrete), 0.8+ (Acoustic panels)
Wall Impedance (Z)	Opposition to sound transmission, depends on density and stiffness.	1.5×10^6 kg·m^{-2}·s^{-1} (Concrete), 5×10^5 kg·m^{-2}·s^{-1} (Brick), 10^4 kg·m^{-2}·s^{-1} (Foam)
Wall Admittance (ζ)	Measure of a wall's ability to accept sound energy, reciprocal of impedance.	0.001 (Concrete), 0.003 (Brick), 0.1 (Foam)

Table B.2: Acoustic properties of walls and their typical values.

- **Resonance Frequency (Panel):**

$$f_r = \frac{60}{d}\sqrt{\frac{E}{\rho(1-\nu^2)}} \tag{B.48}$$

where:d = panel thickness, E = Young's modulus, ρ = density, and ν = Poisson's ratio.

- **Absorption Coefficient:**

$$\alpha = 1 - |R|^2 \tag{B.49}$$

- **Wall Impedance:**

$$Z = \left(\frac{p}{v_n}\right) \tag{B.50}$$

where: v_n denotes the particle component normal to the wall.

- **Wall Admittance:**

$$\zeta = \frac{Z}{\rho_0 * c} \tag{B.51}$$

$$p_i(x,t) = \hat{p}_0 * e^{i*(\omega*t-k*x)} \tag{B.52}$$

$$v_i(x,t) = \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t-k*x)} \tag{B.53}$$

$$p_r(x,t) = R * \hat{p}_0 * e^{i*(\omega*t+k*x)} \tag{B.54}$$

$$v_r(x,t) = -R * \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t+k*x)} \tag{B.55}$$

$$p(0,t) = (1+R) * \hat{p}_0 * e^{i*(\omega*t)} \tag{B.56}$$

$$v(0,t) = (1-R) * \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t)} \tag{B.57}$$

$$Z = \left(\frac{p}{v_n}\right) \rightarrow Z = \rho_0 * c * \frac{1+R}{1-R} \tag{B.58}$$

$$\zeta = \frac{Z}{\rho_0 * c} \rightarrow \zeta = \frac{\rho_0 * c * \left(\frac{1+R}{1-R}\right)}{\rho_0 * c} \rightarrow \zeta = \frac{1+R}{1-R} \tag{B.59}$$

$$R = \frac{\zeta - 1}{\zeta + 1} \tag{B.60}$$

$$\alpha = \frac{4 * Re(\zeta)}{|\zeta|^2 + 2 * Re(\zeta) + 1} \tag{B.61}$$

$$x' = x * cos(\theta) + y * sin(\theta) \tag{B.62}$$

$$p_i(x', t) = \hat{p}_0 * e^{i*(\omega*t - k*(x*cos(\theta) + y*sin(\theta)))} \tag{B.63}$$

$$v_i(x', t) = \frac{\hat{p_0}}{\rho_0 * c} * cos(\theta) * e^{i*(\omega*t - k*(x*cos(\theta) + y*sin(\theta)))} \tag{B.64}$$

$$p_r(x', t) = R * \hat{p}_0 * e^{i*(\omega*t + k*(x*cos(\theta) + y*sin(\theta)))} \tag{B.65}$$

$$v_r(x', t) = -R * \frac{\hat{p_0}}{\rho_0 * c} * e^{i*(\omega*t + k*(x*cos(\theta) + y*sin(\theta)))} \tag{B.66}$$

$$Z = \left(\frac{p}{v_n}\right) \rightarrow Z = \frac{\rho_0 * c}{cos(\theta)} * \frac{1+R}{1-R} \tag{B.67}$$

$$R = \frac{Z * cos(\theta) - \rho_0 * c}{Z * cos(\theta) + \rho_0 * c} = \frac{\zeta * cos(\theta) - 1}{\zeta * cos(\theta) + 1} \tag{B.68}$$

$$\alpha(\theta) = \frac{4 * Re(\zeta) * cos(\theta)}{(|\zeta| * cos(\theta))^2 + 2 * Re(\zeta) * cos(\theta) + 1} \tag{B.69}$$

$$p(x, y) = \hat{p}[1 + |R|^2 + 2 * |R| * cos(2 * k * x * cos(\theta) + \chi]^{1/2} * e^{-i*k*y*(sin\theta)} \tag{B.70}$$

$$c_y = \frac{\omega}{k_y} = \frac{\omega}{k * sin(\theta)} = \frac{c}{sin(\theta)} \tag{B.71}$$

C | Plane Waves

As most of the notation and equations in [7] are equal or at the very least similar to other sources, it has been chosen as the main source for equations, meaning all formulas used in this section can be found in [7] as well, unless otherwise noted. Other sources used for mathematical formulas are [4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. A plane wave simply put is a fraction of a spherical wave small enough that it can be assumed to have no curvature, [36, 9, 13, 14, 4, 5, 7, 8]. Another physical assumption that can be made, is that if a sound wave is present within a circular tube, with a diameter or a rectangular tube with a height/width significantly smaller than the wavelength of said wave, it can be assumed to be planar, see section 3.2 on page 29 for more. It is effectively an orthogonal plane to a vector in a cartesian coordinate system. If that vector is equal to the x-axis, a plane wave can be described by the wave equation (2.2) with:

$$c^2 \frac{\partial^2 p}{\partial x^2} = \frac{\partial^2 p}{\partial t^2} \tag{C.1}$$

With a corresponding general solution given by:

$$p(x,t) = F(c*t-x) + G(c*t+x) \tag{C.2}$$

In equation C.2 the first term represents a plane wave propagating positively along the x-axis and the second term a negative direction. From equations C.1 and C.2 it can also be seen that a constant pressure level is present at each wavefront. By specifying F and G as exponential functions including the imaginary components, the propagation is given as:

$$p(x,t) = \hat{p} * e^{i*k*(c*t-x)} = \hat{p} * e^{i*(\omega*t-k*x)} \tag{C.3}$$

Where $\hat{p}$ is amplitude, k is the propagation constant $k = \frac{\omega}{c}$, ω is the angular frequency, which can be used to obtain the temporal period $T = \frac{2*\pi}{\omega}$. This all connects to several interpretations of the wavelength formula:

$$\lambda = \frac{2*\pi}{k} \leftrightarrow \frac{2*\pi}{\frac{\omega}{c}} \leftrightarrow \frac{2*\pi*c}{\omega} \leftrightarrow \frac{c}{\frac{\omega}{2*\pi}} \leftrightarrow \frac{c}{f} \tag{C.4}$$

Now that amplitude, wavelength, and direction are known, the only missing parameters for expressing the plane wave are frequency and intensity. The frequency is given by $f = \frac{\omega}{2*\pi} * \frac{1}{T}$ with the unit Hz. If equation 2.5 is applied to equation C.2, the only non-vanishing point of the wave is parallel to the x-axis, meaning that sound behaves as longitudinal waves in fluids, and the particle velocity can be found by:

$$v(x,t) = \frac{1}{\rho_0 * c}[F(c*t-x) - G(c*t+x)] \tag{C.5}$$

To obtain the ratio between sound pressure and particle velocity, also known as the characteristic impedance of a medium, the following is used:

$$\frac{p}{v} = \rho_0 * c \tag{C.6}$$

With $\rho_0 * c$ found by table values for ρ_0 and c in the correct medium. For air, the characteristic impedance is:

$$\rho_{0_{air}} * c_{air} = 1.2 \left[\frac{kg}{m^3}\right] * 343 \left[\frac{m}{s}\right] = 414 \left[\frac{kg}{m^2 s}\right] \tag{C.7}$$

Understanding the above relation makes it possible to determine the intensity of each wave. The plane imagined to be orthogonal to the x-axis will have energy present across its entire surface, and the average (denoted by the bar notation) product of the pressure and particle velocity on the surface yields the intensity:

$$I = \bar{pv} = \frac{\bar{p^2}}{\rho_0 * c} \tag{C.8}$$

To find the energy density of a wave, equation C.9 is used:

$$w = \frac{I}{c} = \frac{\bar{p^2}}{\rho_0 * c^2} \tag{C.9}$$

And with that, sufficient information should be available to describe any plane wave.

D | Spherical Waves

As most of the notation and equations in [7] are equal or at the very least similar to other sources, it has been chosen as the main source for equations, meaning all formulas used in this section can be found in [7] as well, unless otherwise noted. Other sources used for mathematical formulas are [4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. The sound wave will now propagate with no boundaries whatsoever, leading to ideal spherical waves. Graphically this is best understood as an inflating ball stretching equally in all directions from an infinitesimally small point source. Due to the nature of spherical waves, a spherical coordinate system will be used, using polar coordinates. A benefit from the spherical symmetry is that pressure is represented as r and is independent of the θ and ϕ angles. To convert the wave equation from cartesian to spherical coordinates, the laplacian of a function $p(r, \theta, \phi)$has to expressed in spherical coordinates as well:

$$\Delta p = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial p}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\,\frac{\partial p}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2 p}{\partial\phi^2}. \tag{D.1}$$

As the acoustical properties of a spherical wave are equal across all θ and ϕ angles, it's description can be made with only r and t, simplifying the laplacian to:

$$\Delta p = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial p}{\partial r}\right) \tag{D.2}$$

The simplified laplacian should now be expanded to:

$$\Delta p = \frac{\partial^2 p}{\partial r^2} + \frac{2}{r}\frac{\partial p}{\partial r} \tag{D.3}$$

And is now ready to be inputted into the wave equation:

$$c^2\left(\frac{\partial^2 p}{\partial r^2} + \frac{2}{r}\frac{\partial p}{\partial r}\right) = \frac{\partial^2 p}{\partial t^2} \tag{D.4}$$

By dividing with c^2 on both sides, the spherical wave equation is found:

$$\frac{\partial^2 p}{\partial r^2} + \frac{2}{r} * \frac{\partial p}{\partial r} = \frac{1}{c^2} * \frac{\partial^2 p}{\partial t^2} \tag{D.5}$$

A simple solution to the spherical wave equation is given in equation D.6, representing a spherical wave with the volume velocity Q, being the rate in $\frac{m^3}{s}$ of "liquid" being expelled by the sound source located at $r = 0$. The dot above Q denotes differentation w.r.t. time (t).

$$p(r, t) = \frac{\rho_0}{4 * \pi * r} * \dot{Q}\left(t - \frac{r}{c}\right) \tag{D.6}$$

As for plane waves, spherical waves have a non-vanishing component of their particle velocity. Instead of being along the x-axis, the non-vanishing point for a spherical wave is radial, and can be calculated by applying equation 2.5 to D.6, yielding:

$$v_r = \frac{1}{4 * \pi * r^2} \left[Q\left(t - \frac{r}{c}\right) + \frac{r}{c} * \dot{Q}\left(t - \frac{r}{c}\right) \right] \tag{D.7}$$

To find the particle velocity of a spherical wave, equation D.8 is used. It shows that the sound pressure and velocity in a spherical wave is inversely proportional to the size r and frequency $\omega = k * c$. For distances large in comparison with the wavelength, the ratio $\frac{p}{v}$ tends asymptotically to $\rho_0 c$.

$$v_r = \frac{p}{\rho_0 * c} * \left(1 + \frac{1}{i * k * r}\right) \tag{D.8}$$

E | Octave Bands

When examining sound and acoustic properties, it is often done in octave bands or fractions thereof, [69, 70, 71, 72]. Octaves emerge from the world of music, given at the interval between two musical pitches, double or half the frequency. Octave values loosely follows 2s complement, with some rounding error, [70]. In figure E.1 the most common centerfrequencies can be seen to broadly follow 2s complement. The audible spectrum from 20Hz-20kHz is often divided into $\tilde{1}1$ octave bands, with the 7th band being set to 1kHz. Mathematically the octave bands are given by equation E.1, where f_c is set to be 1kHz and n is the octave band number. The upper and lower frequency band cutoff can be found by equation E.2, where f_c is set to the relevant centerfrequency.

$$Lower : f_c(n-1) = \frac{f_c(n)}{2} \qquad Upper : f_c(n+1) = 2 * f_c(n) \tag{E.1}$$

$$f_c = \sqrt{2} * f_{min} = \frac{f_{max}}{\sqrt{2}} \tag{E.2}$$

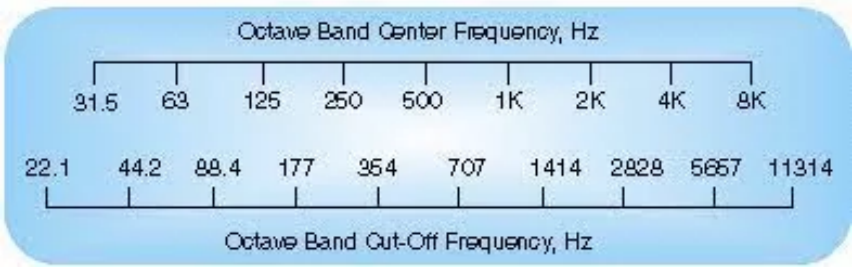


Figure E.1: 1/1 octave bands, with centerfrequencies at the top, and cutoff-frequencies at the bottom, [71].

To gain greater resolution most measurements within room acoustics are done in fractional octave bands, often $\frac{1}{3}$. When calculating the $\frac{1}{3}$ octave bands, equation E.1 is modified to:

$$Lower : f_c(n-1) = \frac{f_c(n)}{2^{\frac{1}{3}}} \qquad Upper : f_c(n+1) = 2^{\frac{1}{3}} * f_c(n) \tag{E.3}$$

And instead of $\tilde{1}1$ octave bands, $\tilde{3}1$ is used instead, making 1khz the 19th octave bands centerfrequency. As the octave bands have gotten singificantly smaller, the upper and lower cutoff is adjusted accordingly to:

$$f_{c\frac{1}{3}} = \frac{1}{3} * \sqrt{2} * f_{min\frac{1}{3}} = \frac{1}{3} * \frac{f_{max\frac{1}{3}}}{\sqrt{2}} \tag{E.4}$$

In table E.1 on page 123 the hearing spectrum is mapped into $\frac{1}{1}$ and $\frac{1}{3}$ octave bands.

Octave Bands			1/3 Octave Bands		
Lower Band Limit (Hz)	**Center Frequency (Hz)**	**Upper Band Limit (Hz)**	**Lower Band Limit (Hz)**	**Center Frequency (Hz)**	**Upper Band Limit (Hz)**
			14.1	16	17.8
11	16	22	17.8	20	22.4
			22.4	25	28.2
			28.2	31.5	35.5
22	31.5	44	35.5	40	44.7
			44.7	50	56.2
			56.2	63	70.8
44	63	88	70.8	80	89.1
			89.1	100	112
			112	125	141
88	125	177	141	160	178
			178	200	224
			224	250	282
177	250	355	282	315	355
			355	400	447
			447	500	562
355	500	710	562	630	708
			708	800	891
			891	1000	1122
710	1000	1420	1122	1250	1413
			1413	1600	1778
			1778	2000	2239
1420	2000	2840	2239	2500	2818
			2818	3150	3548
			3548	4000	4467
2840	4000	5680	4467	5000	5623
			5623	6300	7079
			7079	8000	8913
5680	8000	11360	8913	10000	11220
			11220	12500	14130
			14130	16000	17780
11360	16000	22720	17780	20000	22390

Table E.1: Octave Bands and 1/3 Octave Bands, [70].

F | Standardized Renard Numbers

Standard frequencies (c/s)	American 1/3-octave band number
100	20
125	21
160	22
200	23
250	24
315	25
400	26
500	27
630	28
800	29
1000	30
1250	31
1600	32
2000	33
2500	34
3150	35
4000	36
5000	37
6300	38

40-series	20-series	10-series	5-series	Exact Value	Mantissa
1.00	1.00	1.00	1.00	10000	000
1.06				10593	025
1.12	1.12			11220	050
1.18				11885	075
1.25	1.25	1.25		12589	100
1.32				13335	125
1.40	1.40			14125	150
1.50				14962	175
1.60	1.60	1.60	1.60	15849	200
1.70				16788	225
1.80	1.80			17783	250
1.90				18836	275
2.00	2.00	2.00		19953	300
2.12				21135	325
2.24	2.24			22387	350
2.36				23714	375
2.50	2.50	2.50	2.50	25119	400
2.65				26607	425
2.80	2.80			28184	450
3.00				29854	475
3.15	3.15	3.15		31623	500
3.35				33497	525
3.55	3.55			35481	550
3.75				37594	575
4.00	4.00	4.00	4.00	39811	600
4.25				42170	625
4.50	4.50			44668	650
4.75				47315	675
5.00	5.00	5.00		50119	700
5.30				53088	725
5.60	5.60			56234	750
6.00				59566	775
6.30	6.30	6.30	6.30	63096	800
6.70				66834	825
7.10	7.10			70795	850
7.50				74989	875
8.00	8.00	8.00		79433	900
8.50				84140	925
9.00	9.00			89125	950
9.50				94406	975

Table F.1: Standard frequencies which should be chosen when taking measurements in the apparatus. The frequencies are based on the R_{10} series of the international "Preferred Numbers". A complete table of these so-called "standardized Renard numbers" dividing a decade into 10 equal logarithmic steps is shown at the right, [32].

G | Maximum Length LFSR Lookup Table

Bits (n)	Feedback Polynomial	Taps	Taps (hex)	Period
2	$x^2 + x + 1$	11	0x3	3
3	$x^3 + x^2 + 1$	110	0x6	7
4	$x^4 + x^3 + 1$	1100	0xC	15
5	$x^5 + x^3 + 1$	10100	0x14	31
6	$x^6 + x^5 + 1$	110000	0x30	63
7	$x^7 + x^6 + 1$	1100000	0x60	127
8	$x^8 + x^6 + x^5 + x^4 + 1$	10111000	0xB8	255
9	$x^9 + x^5 + 1$	100010000	0x110	511
10	$x^{10} + x^7 + 1$	1001000000	0x240	1,023
11	$x^{11} + x^9 + 1$	10100000000	0x500	2,047
12	$x^{12} + x^{11} + x^{10} + x^4 + 1$	111000001000	0xE08	4,095
13	$x^{13} + x^{12} + x^{11} + x^8 + 1$	1110010000000	0x1C80	8,191
14	$x^{14} + x^{13} + x^{12} + x^2 + 1$	11100000000010	0x3802	16,383
15	$x^{15} + x^{14} + 1$	110000000000000	0x6000	32,767
16	$x^{16} + x^{15} + x^{13} + x^4 + 1$	1101000000001000	0xD008	65,535
17	$x^{17} + x^{14} + 1$	10010000000000000	0x12000	131,071
18	$x^{18} + x^{11} + 1$	100000010000000000	0x20400	262,143
19	$x^{19} + x^{18} + x^{17} + x^{14} + 1$	1110010000000000000	0x72000	524,287
20	$x^{20} + x^{17} + 1$	10010000000000000000	0x90000	1,048,575
21	$x^{21} + x^{19} + 1$	10100000000000000000	0x140000	2,097,151
22	$x^{22} + x^{21} + 1$	11000000000000000000000	0x300000	4,194,303
23	$x^{23} + x^{18} + 1$	10000100000000000000000	0x420000	8,388,607
24	$x^{24} + x^{23} + x^{22} + x^{17} + 1$	11100001000000000000000	0xE10000	16,777,215

Table G.1: Feedback Polynomials, Taps, and Periods for Various Bit-Lengths, [48].

H Schematics For Construction of the Impedance Tube

I MLS Test Outputs

I.1 MLS Generation Time Averages

Bitnumber	Run 1	Run 2	Run 3	Run 4	Run 5	Average	Converted to ms	Converted to s	Converted to minutes	Converter to hours
2	1	1	1	1	1	1	0.001	1.00E-06	1.66667E-08	2.77778E-10
3	2	2	2	1	1	1.6	0.0016	1.60E-06	2.66667E-08	4.44444E-10
4	2	2	2	1	1	1.6	0.0016	1.60E-06	2.66667E-08	4.44444E-10
5	3	3	3	2	2	2.6	0.0026	2.60E-06	4.33333E-08	7.22222E-10
6	5	5	5	4	4	4.6	0.0046	4.60E-06	7.66667E-08	1.27778E-09
7	8	8	10	8	8	8.4	0.0084	8.40E-06	0.00000014	2.33333E-09
8	15	15	15	14	14	14.6	0.0146	1.46E-05	2.43333E-07	4.05556E-09
9	28	28	28	27	27	27.6	0.0276	2.76E-05	0.00000046	7.66667E-09
10	53	53	53	53	53	53	0.053	5.30E-05	8.83333E-07	1.47222E-08
11	106	108	106	106	106	106.4	0.1064	1.06E-04	1.77333E-06	2.95556E-08
12	208	209	208	208	208	208.2	0.2082	2.08E-04	0.00000347	5.78333E-08
13	419	418	418	419	418	418.4	0.4184	4.18E-04	6.97333E-06	1.16222E-07
14	140682	139850	142186	141986	141048	141150.4	141.1504	1.41E-01	0.002352507	3.92084E-05
15	512195	512915	512195	512196	512195	512339.2	512.3392	5.12E-01	0.008538987	0.000142316
16	1255234	1255233	1255234	1255234	1255234	1255233.8	1255.2338	1.26E+00	0.020920563	0.000348676
17	2741311	2741310	2741311	2741311	2741311	2741310.8	2741.3108	2.74E+00	0.045688513	0.000761475
18	5713464	5713465	5713465	5713466	5713465	5713465	5713.465	5.71E+00	0.095224417	0.001587074
19	11657773	11657773	11657773	11657773	11657773	11657773	11657.773	1.17E+01	0.194296217	0.00323827
20	23546389	23546390	23546389	23546390	23546390	23546390	23546.3896	2.35E+01	0.392439827	0.006540664
21	47323624	47323664	47323624	47323624	47323624	47323632	47323.632	4.73E+01	0.7887272	0.013145453
22	94878091	94878090	94878091	94878091	94878091	94878091	94878.0908	9.49E+01	1.581301513	0.026355025
23	189987025	189987025	189987025	189987025	189987026	189987025	189987.0252	1.90E+02	3.16645042	0.052774174
24	380204892	380204893	380204892	380204893	380204894	380204893	380204.8928	3.80E+02	6.336748213	0.10561247
25	760636942					760636942	760636.942	7.61E+02	12.67728237	0.211288039
26	1521511257					1.522E+09	1521511.257	1.52E+03	25.35852095	0.422642016
27	3043253279					3.043E+09	3043253.279	3.04E+03	50.72088798	0.845348133
28	6086624914					6.087E+09	6086624.914	6.09E+03	101.4437486	1.690729143
29	1.2173E+10					1.217E+10	12173368.18	1.22E+04	202.8894697	3.381491162
30	2.4347E+10					2.435E+10	24346854.72	2.43E+04	405.7809121	6.763015201
31	4.8694E+10					4.869E+10	48693827.8	4.87E+04	811.5637967	13.52606328
32	9.7388E+10					9.739E+10	97387773.96	9.74E+04	1623.129566	27.05215943

Figure I.1: Average time taken to generate 2-32 bit MLSs. As mentioned in 5.2.2, 25-32 bit has only been iterated once due to the exceptionally long execution time.

I.2 Correct Teensy Generated MLS - MLSChecker Results and Plots

Sequence length: 15 samples
Detected register size: 4 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 15 samples
Register size (n) : 4 bits
Detected cyclic shift : 0
Autocorrelation peak : 15.000000 (ideal = 15)
Max sidelobe deviation: 0.000000

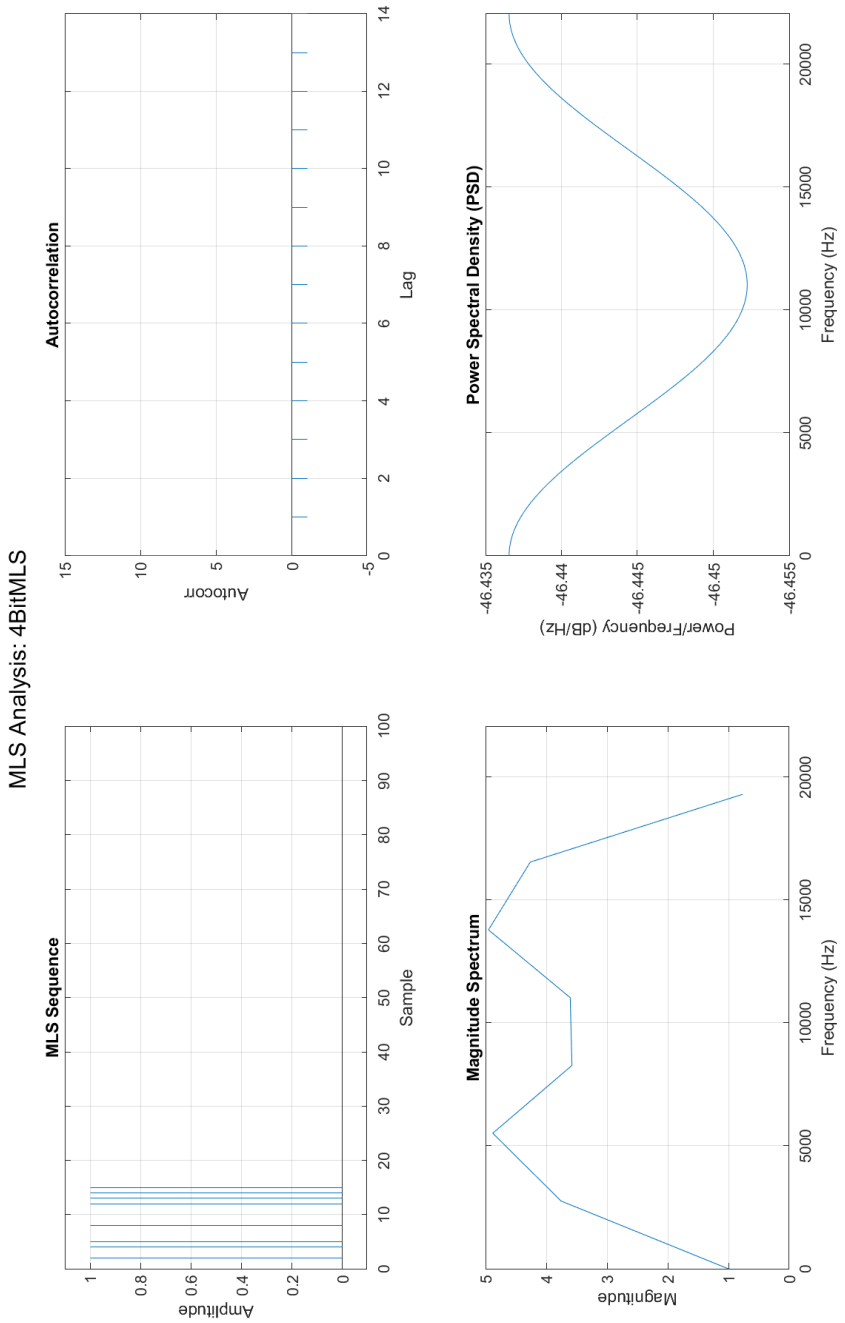


Figure I.2: Plots related to the 4-bit MLS sequence generated on the Teensy.

Sequence length: 31 samples
Detected register size: 5 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 31 samples
Register size (n) : 5 bits
Detected cyclic shift : 0
Autocorrelation peak : 31.000000 (ideal = 31) Max sidelobe deviation: 0.000000

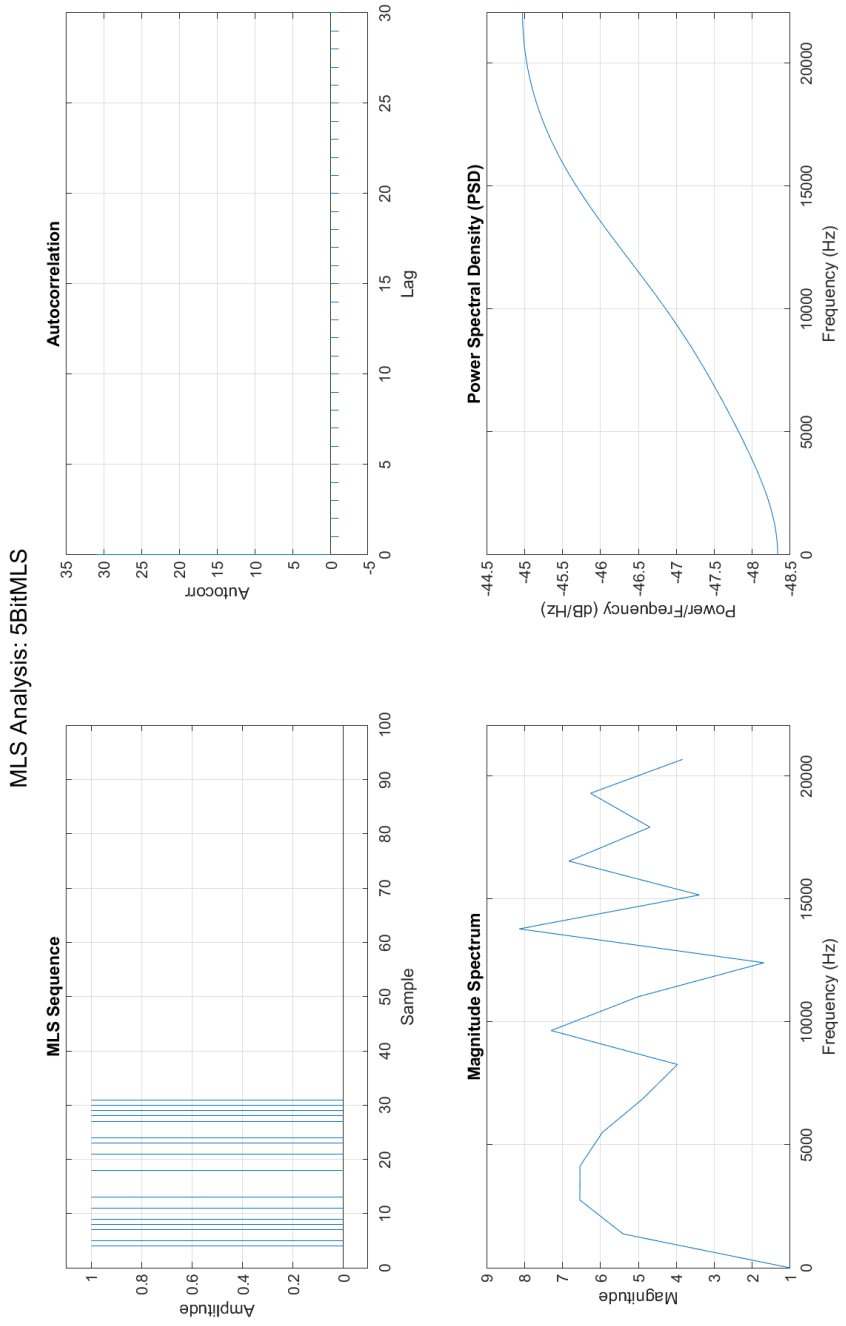


Figure I.3: Plots related to the 5-bit MLS sequence generated on the Teensy.

Sequence length: 63 samples
Detected register size: 6 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 63 samples
Register size (n) : 6 bits
Detected cyclic shift : 0
Autocorrelation peak : 63.000000 (ideal = 63)
Max sidelobe deviation: 0.000000

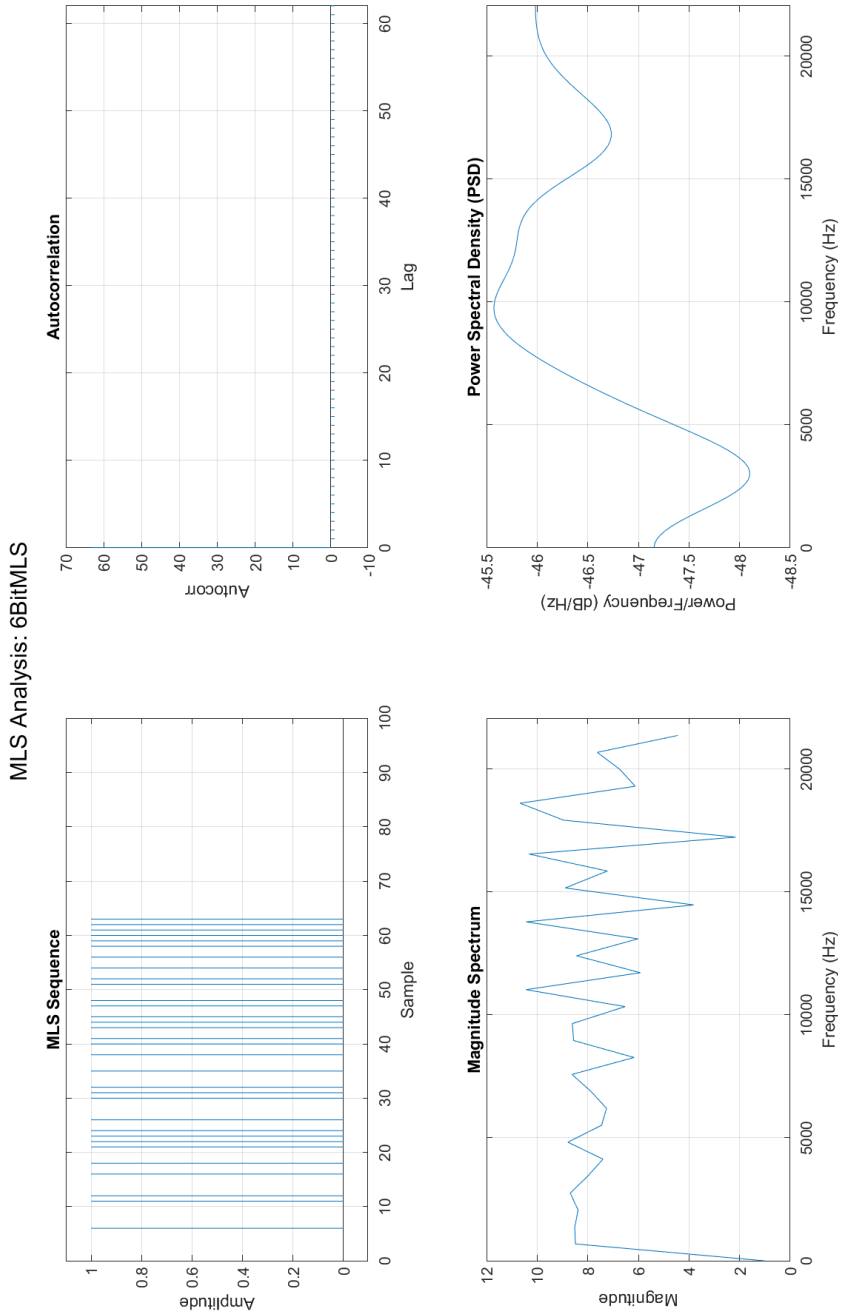


Figure I.4: Plots related to the 6-bit MLS sequence generated on the Teensy.

Sequence length: 127 samples
Detected register size: 7 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 127 samples
Register size (n) : 7 bits
Detected cyclic shift : 0
Autocorrelation peak : 127.000000 (ideal = 127)
Max sidelobe deviation: 0.000000

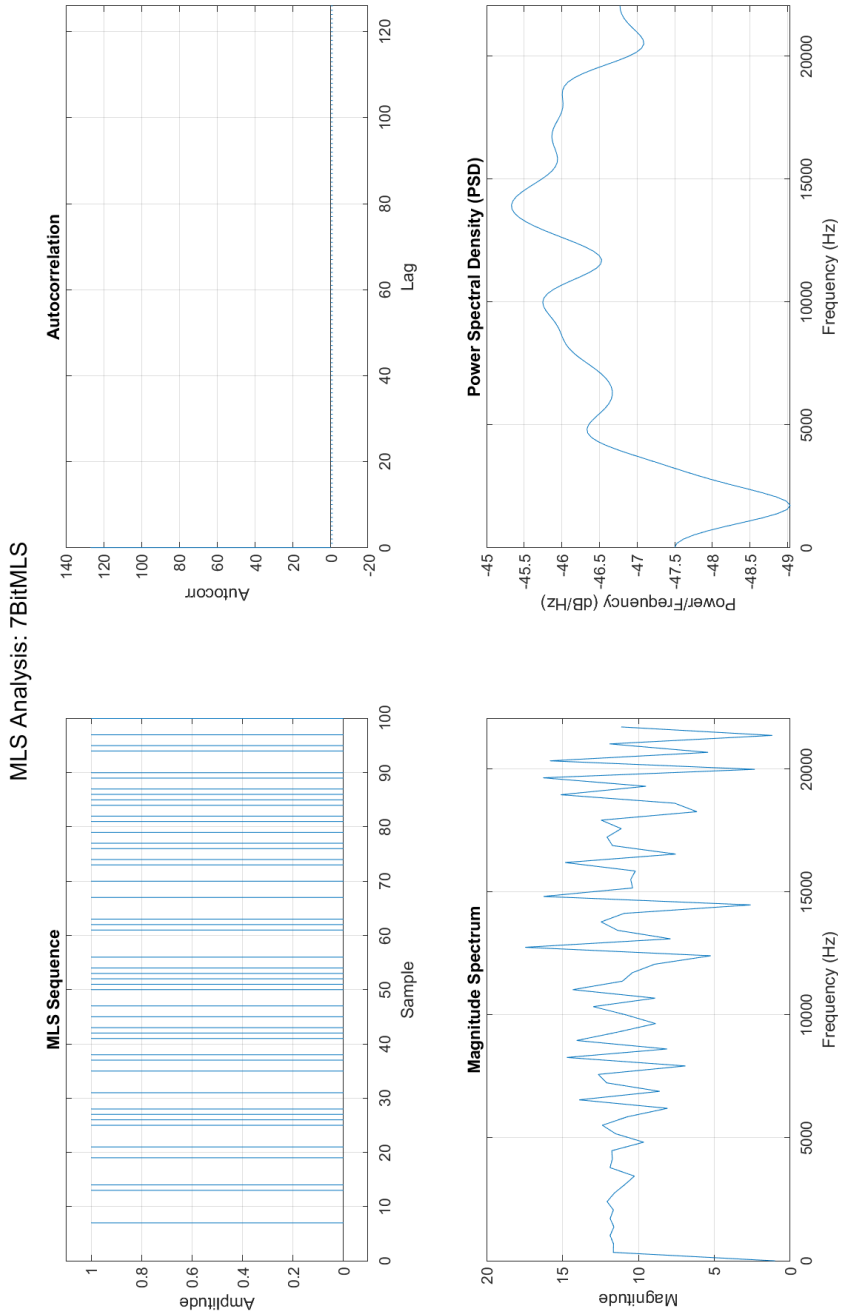


Figure I.5: Plots related to the 7-bit MLS sequence generated on the Teensy.

Sequence length: 255 samples
Detected register size: 8 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 255 samples
Register size (n) : 8 bits
Detected cyclic shift : 0
Autocorrelation peak : 255.000000 (ideal = 255)
Max sidelobe deviation: 0.000000

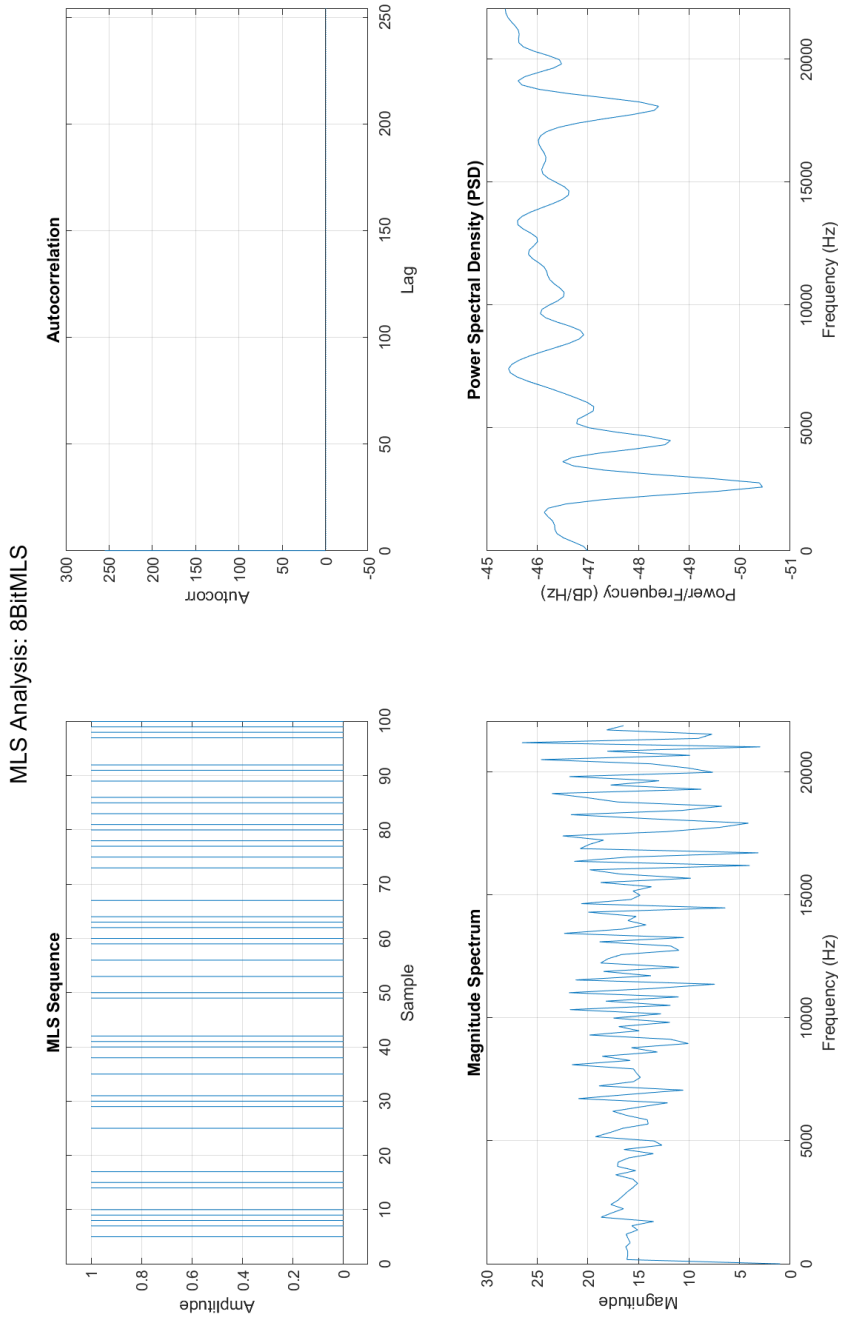


Figure I.6: Plots related to the 8-bit MLS sequence generated on the Teensy.

Sequence length: 511 samples
Detected register size: 9 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 511 samples
Register size (n) : 9 bits
Detected cyclic shift : 0
Autocorrelation peak : 511.000000 (ideal = 511)
Max sidelobe deviation: 0.000000

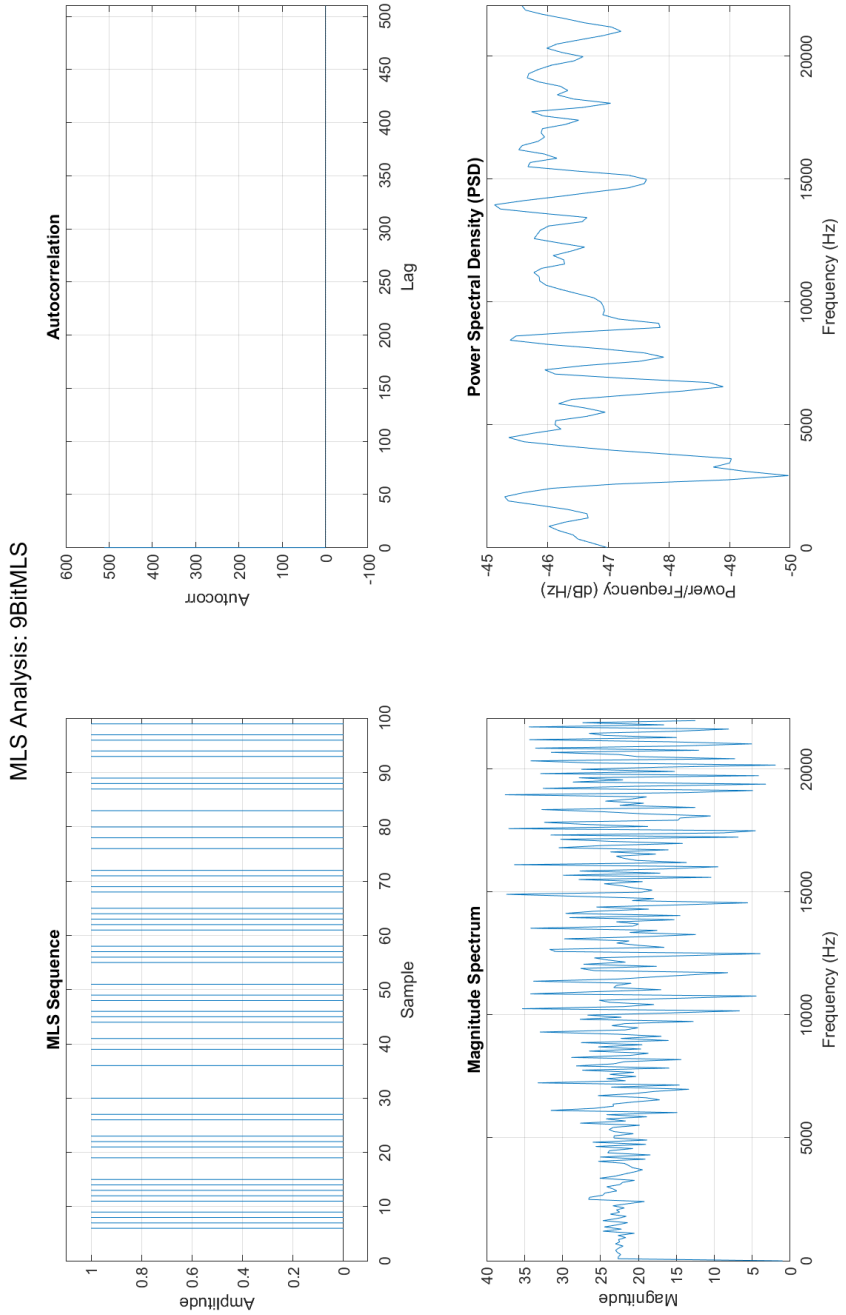


Figure I.7: Plots related to the 9-bit MLS sequence generated on the Teensy.

Sequence length: 1023 samples
Detected register size: 10 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 1023 samples
Register size (n) : 10 bits
Detected cyclic shift : 0
Autocorrelation peak : 1023.000000 (ideal = 1023)
Max sidelobe deviation: 0.000000

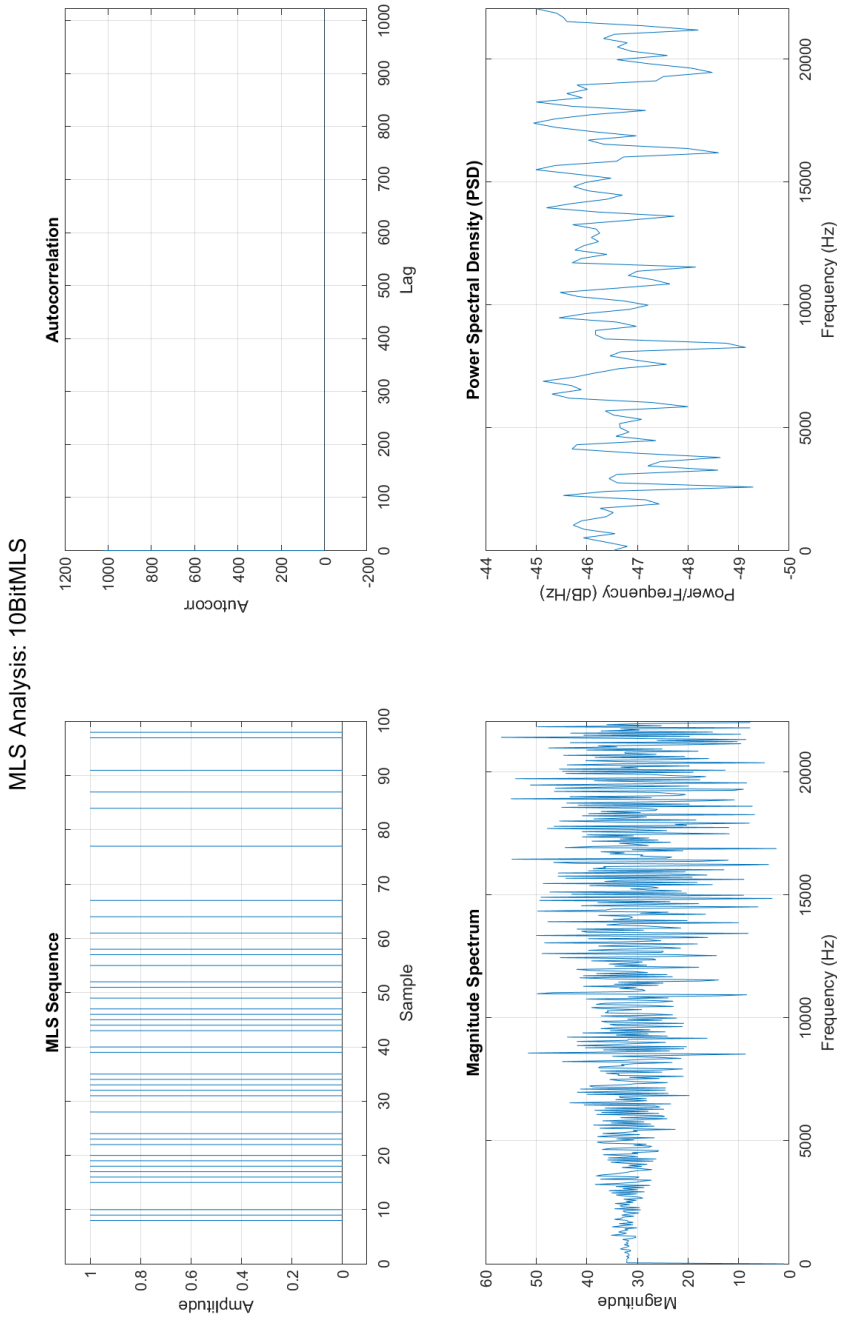


Figure I.8: Plots related to the 10-bit MLS sequence generated on the Teensy.

Sequence length: 2047 samples
Detected register size: 11 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 2047 samples
Register size (n) : 11 bits
Detected cyclic shift : 0
Autocorrelation peak : 2047.000000 (ideal = 2047)
Max sidelobe deviation: 0.000000

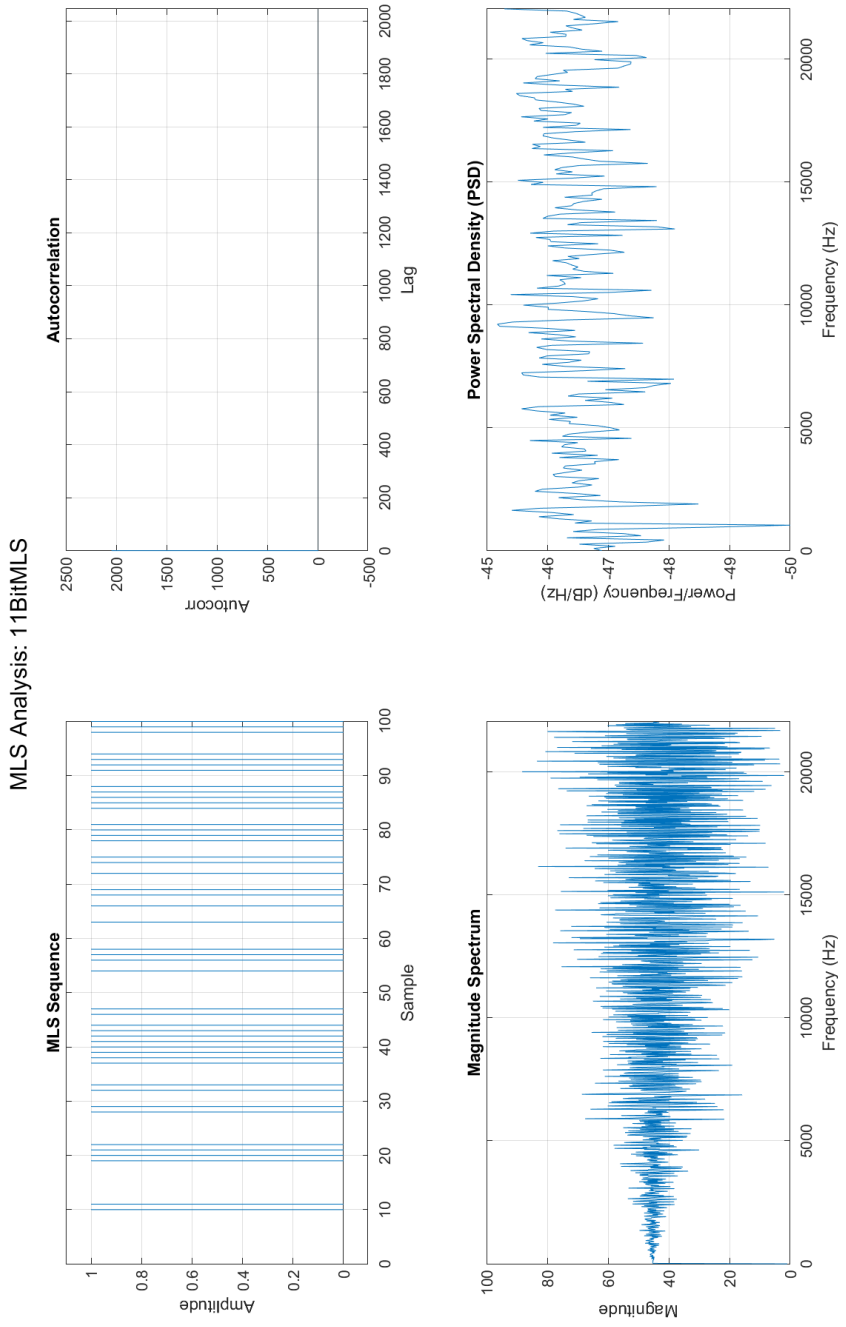


Figure I.9: Plots related to the 11-bit MLS sequence generated on the Teensy.

Sequence length: 4095 samples
Detected register size: 12 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 4095 samples
Register size (n) : 12 bits
Detected cyclic shift : 0
Autocorrelation peak : 4095.000000 (ideal = 4095)
Max sidelobe deviation: 0.000000

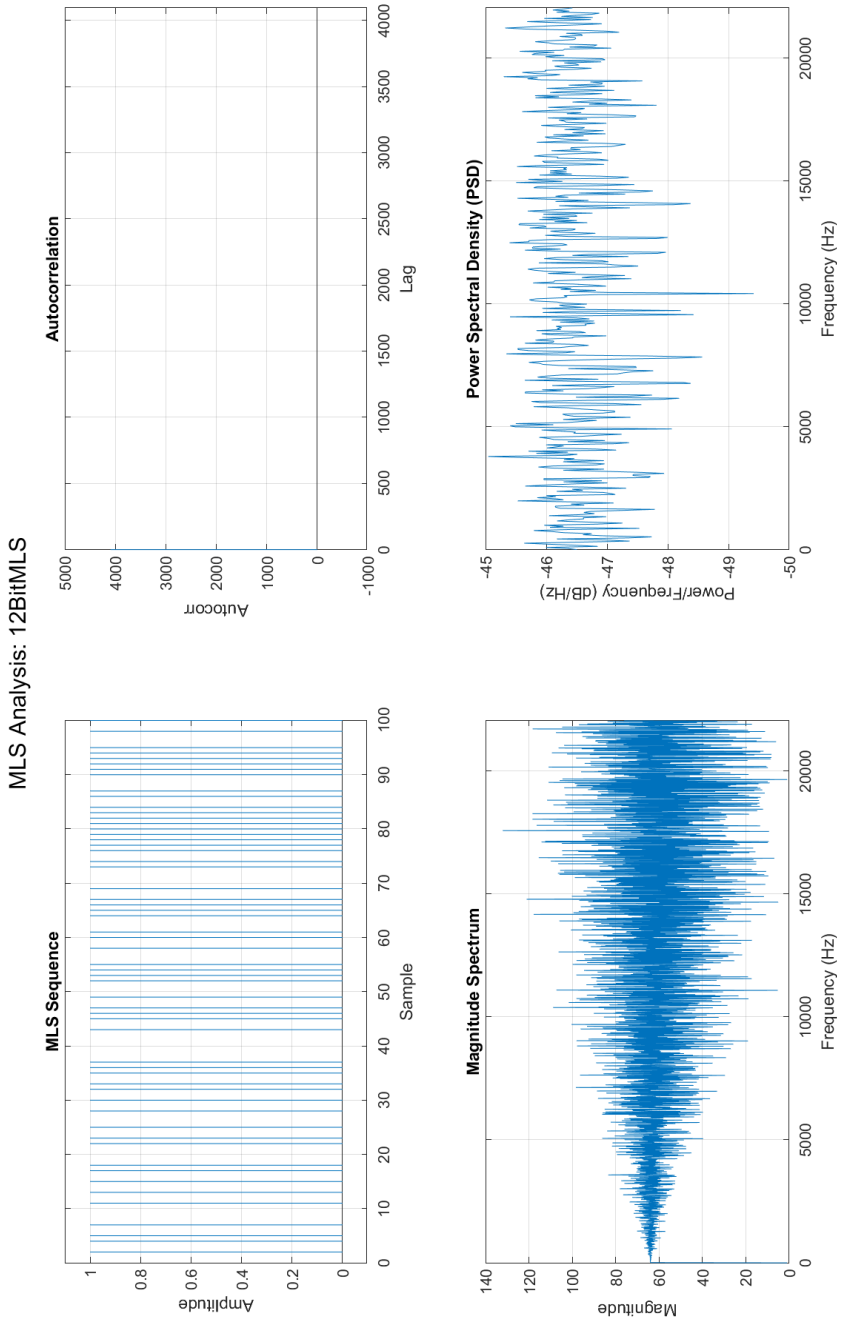


Figure I.10: Plots related to the 12-bit MLS sequence generated on the Teensy.

Sequence length: 8191 samples
Detected register size: 13 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 8191 samples
Register size (n) : 13 bits
Detected cyclic shift : 0
Autocorrelation peak : 8191.000000 (ideal = 8191) Max sidelobe deviation: 0.000000

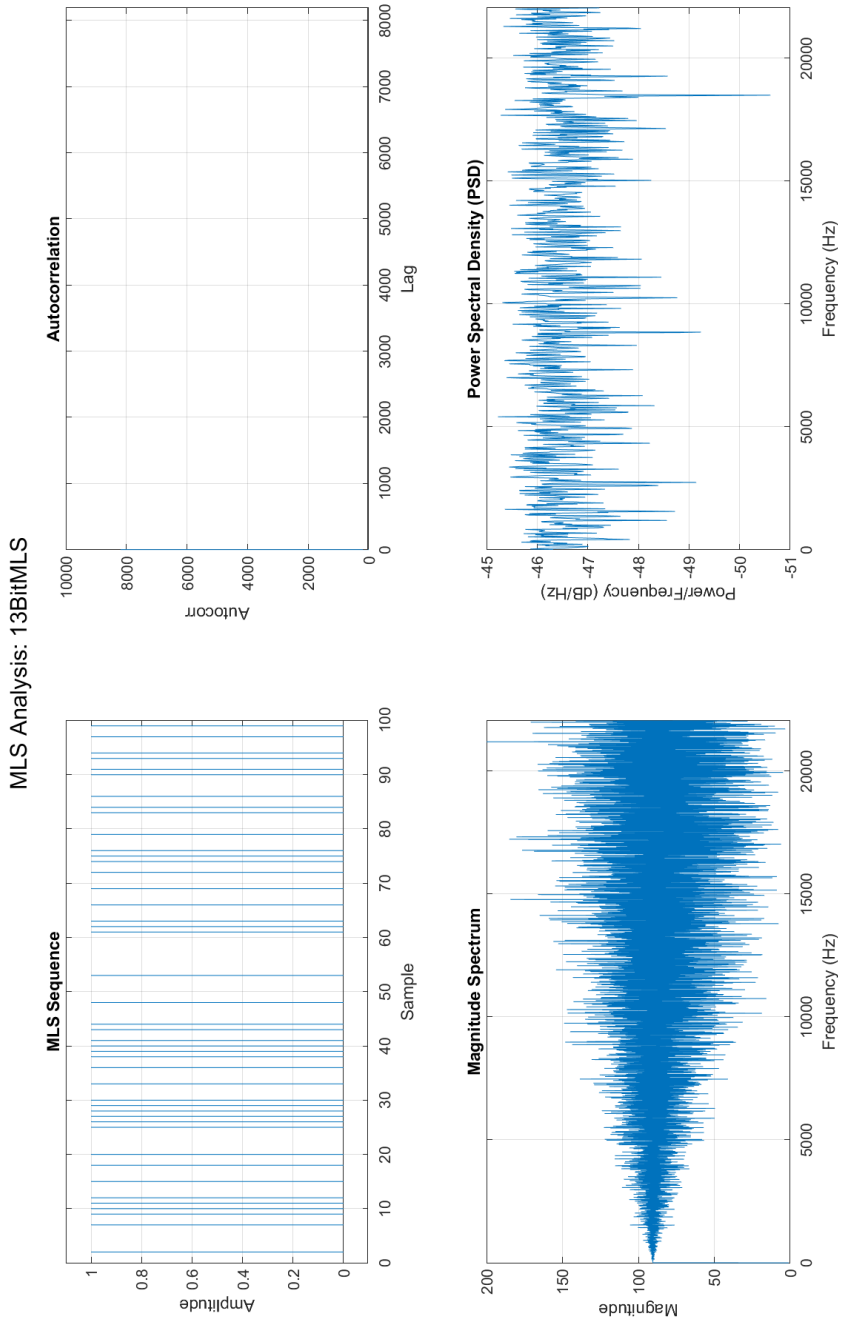


Figure I.11: Plots related to the 13-bit MLS sequence generated on the Teensy.

Sequence length: 16383 samples
Detected register size: 14 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 16383 samples
Register size (n) : 14 bits
Detected cyclic shift : 0
Autocorrelation peak : 16383.000000 (ideal = 16383)
Max sidelobe deviation: 0.000000

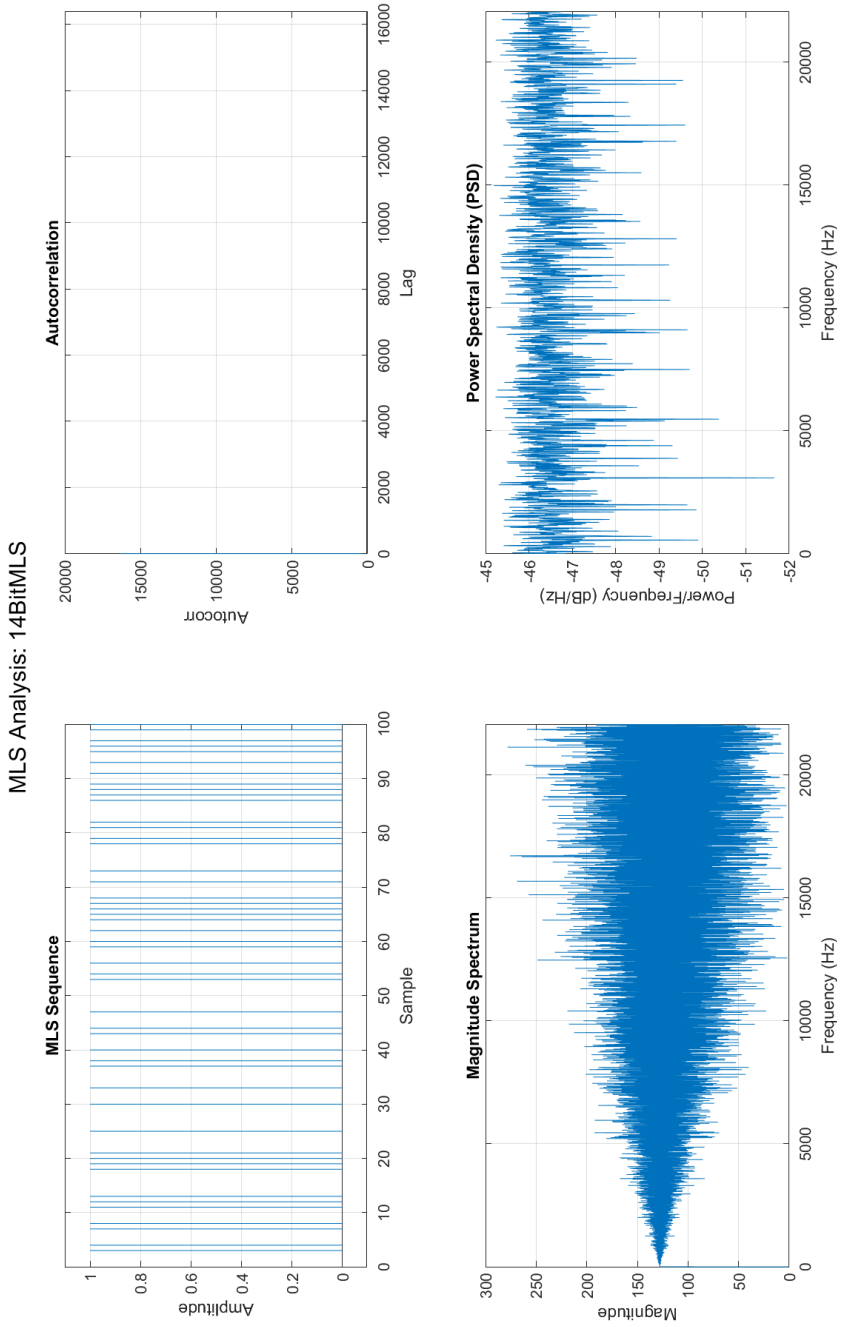


Figure I.12: Plots related to the 14-bit MLS sequence generated on the Teensy.

Sequence length: 32767 samples
Detected register size: 15 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 32767 samples
Register size (n) : 15 bits
Detected cyclic shift : 0
Autocorrelation peak : 32767.000000 (ideal = 32767)
Max sidelobe deviation: 0.000000

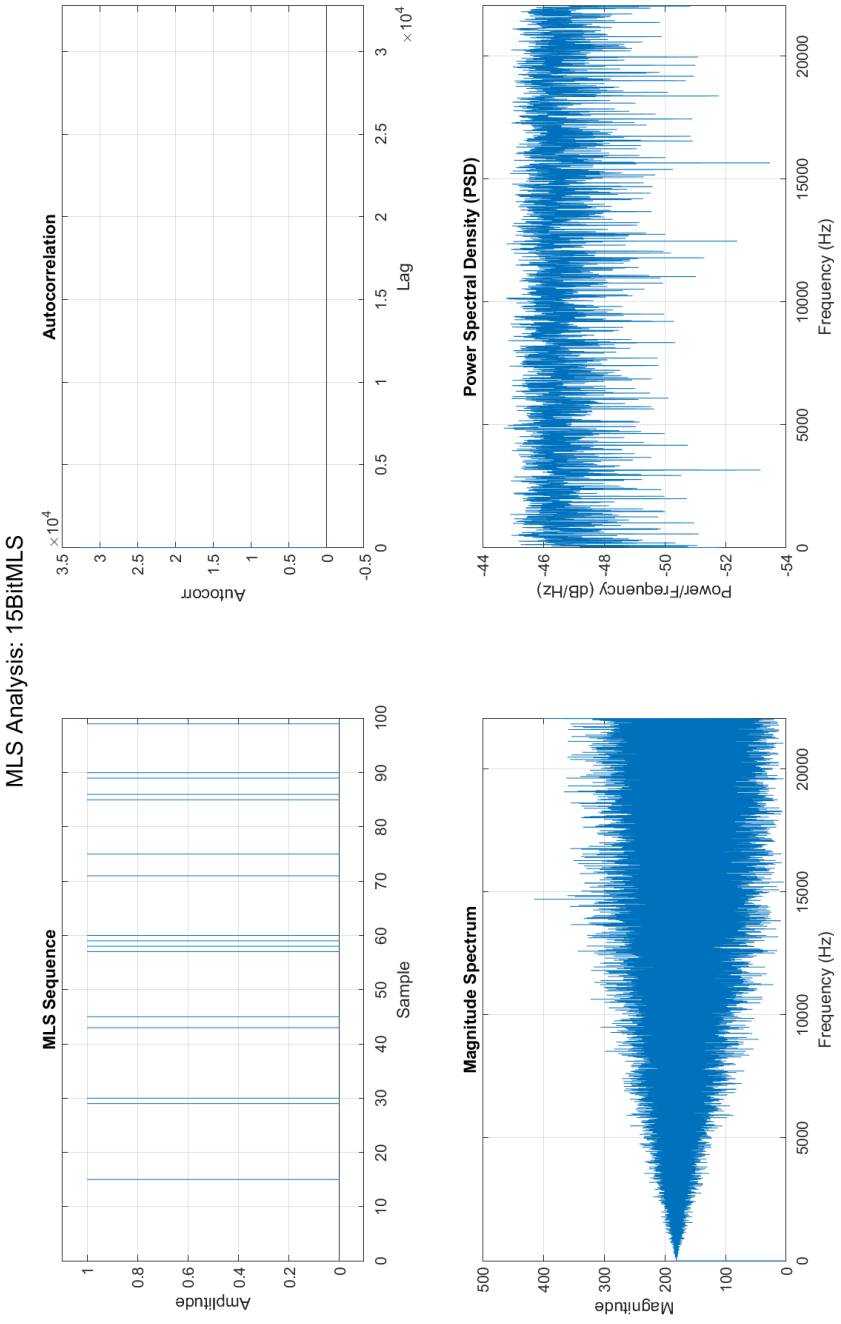


Figure I.13: Plots related to the 15-bit MLS sequence generated on the Teensy.

Sequence length: 65535 samples
Detected register size: 16 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 65535 samples
Register size (n) : 16 bits
Detected cyclic shift : 0
Autocorrelation peak : 65535.000000 (ideal = 65535)
Max sidelobe deviation: 0.000000

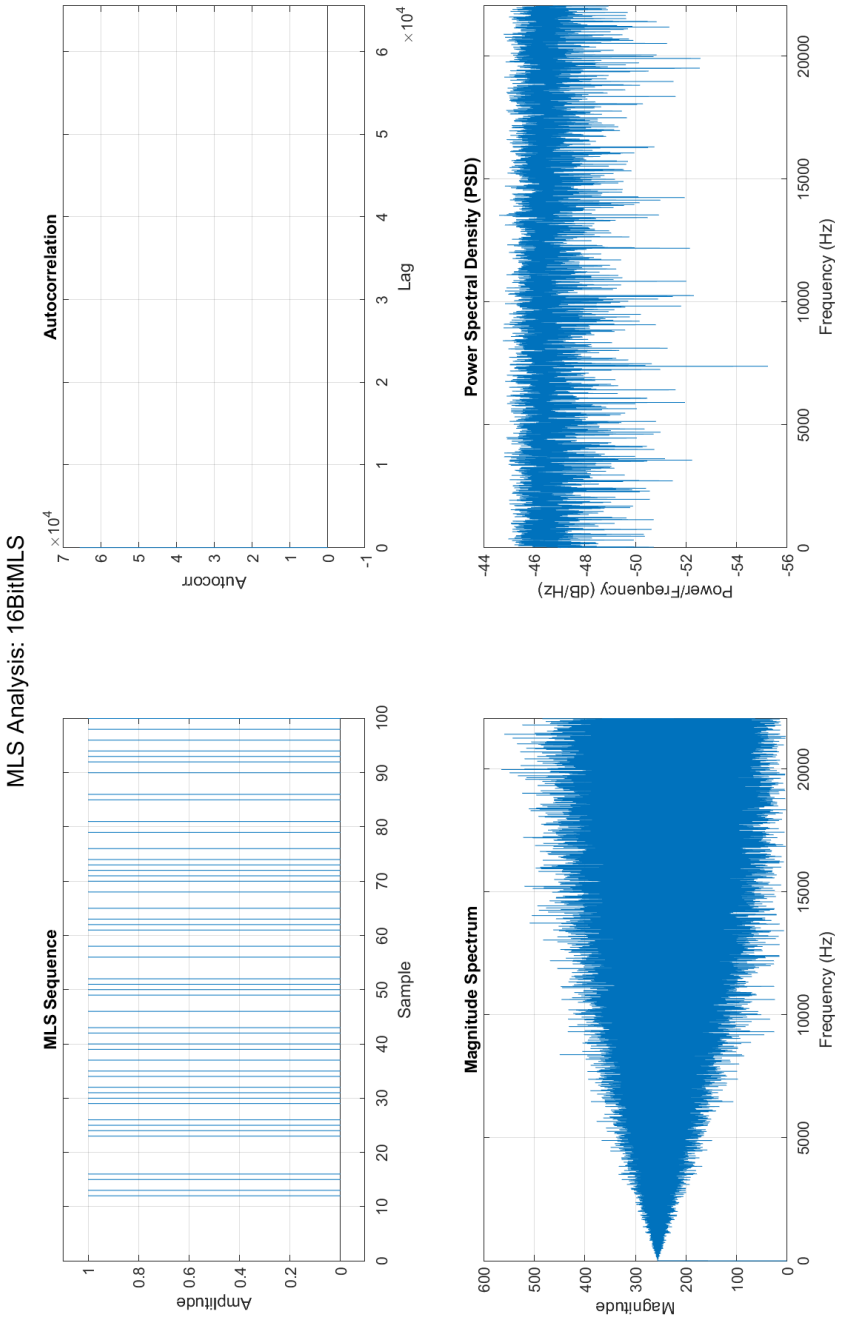


Figure I.14: Plots related to the 16-bit MLS sequence generated on the Teensy.

Sequence length: 131071 samples
Detected register size: 17 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 131071 samples
Register size (n) : 17 bits
Detected cyclic shift : 0
Autocorrelation peak : 131071.000000 (ideal = 131071) Max sidelobe deviation: 0.000000

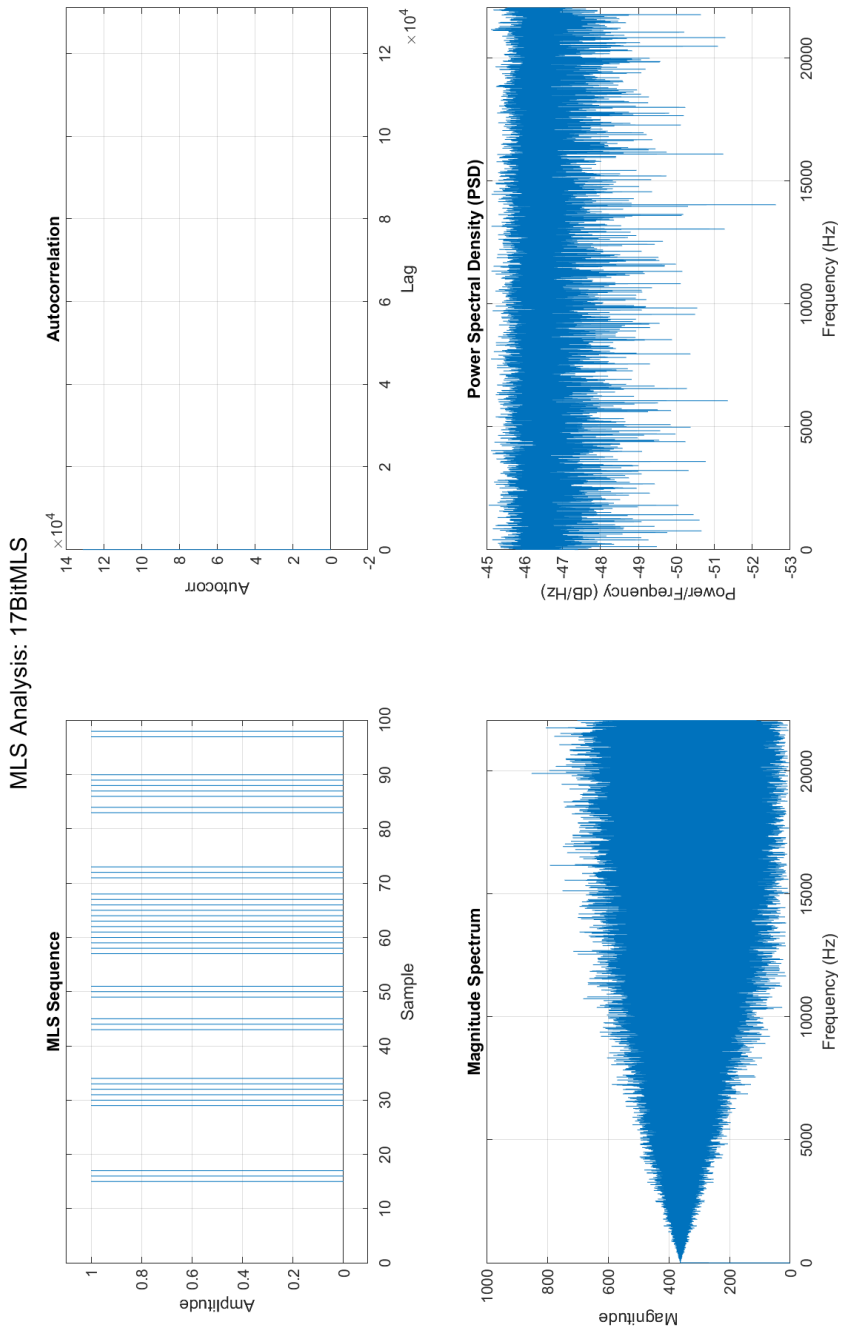


Figure I.15: Plots related to the 17-bit MLS sequence generated on the Teensy.

Sequence length: 262143 samples
Detected register size: 18 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 262143 samples
Register size (n) : 18 bits
Detected cyclic shift : 0
Autocorrelation peak : 262143.000000 (ideal = 262143)
Max sidelobe deviation: 0.000000

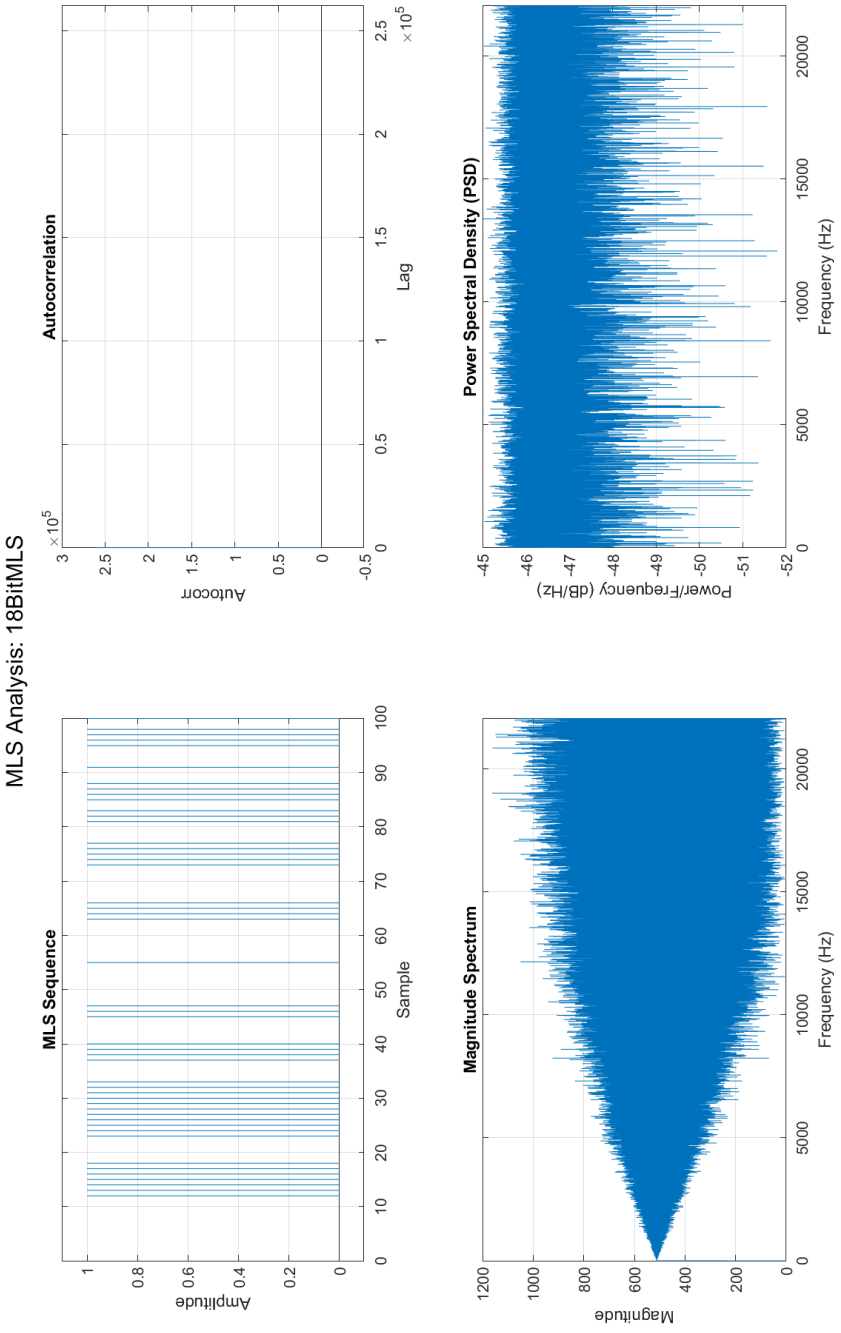


Figure I.16: Plots related to the 18-bit MLS sequence generated on the Teensy.

Sequence length: 524287 samples
Detected register size: 19 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 524287 samples
Register size (n) : 19 bits
Detected cyclic shift : 0
Autocorrelation peak : 524287.000000 (ideal = 524287)
Max sidelobe deviation: 0.000000

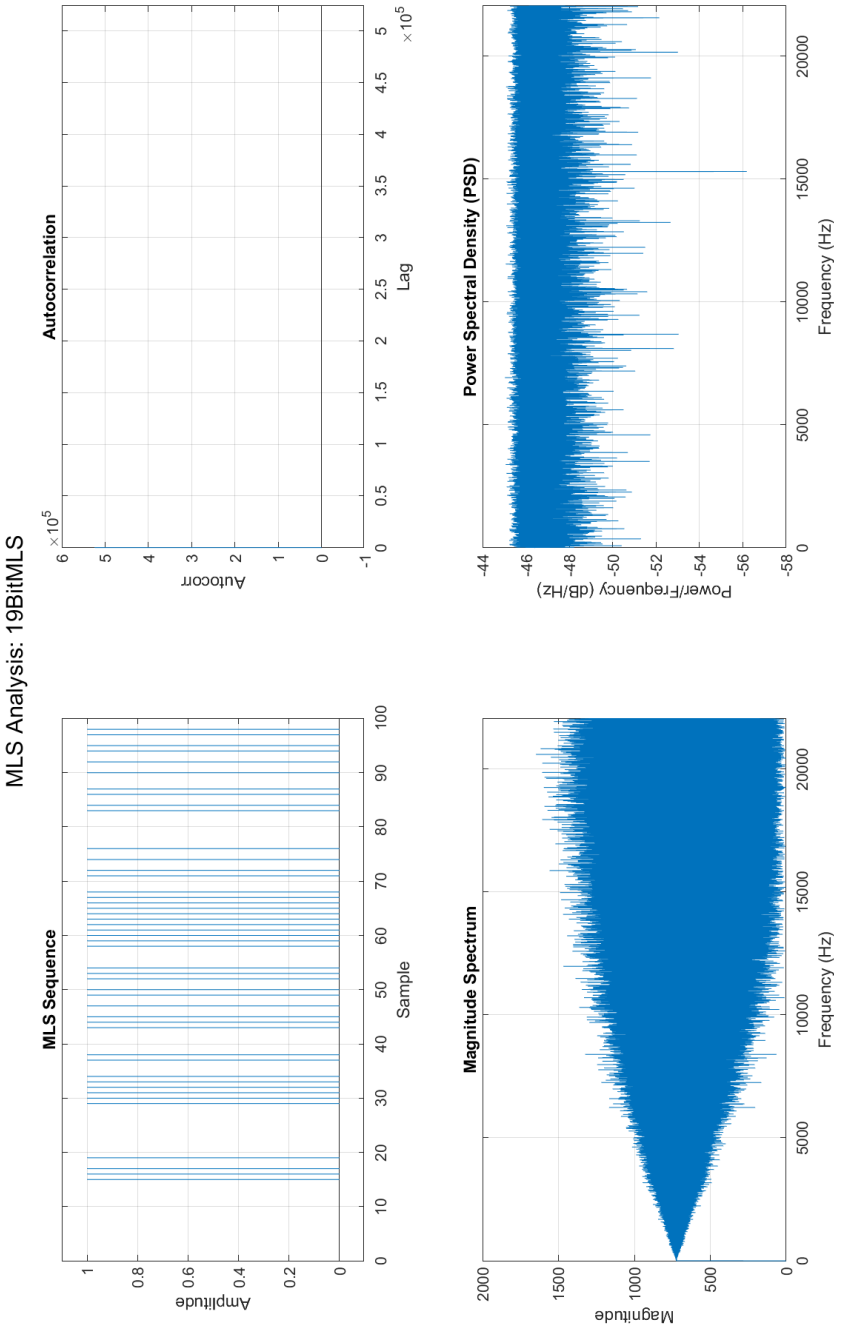


Figure I.17: Plots related to the 19-bit MLS sequence generated on the Teensy.

Sequence length: 1048575 samples
Detected register size: 20 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 1048575 samples
Register size (n) : 20 bits
Detected cyclic shift : 0
Autocorrelation peak : 1048575.000000 (ideal = 1048575)
Max sidelobe deviation: 0.000000

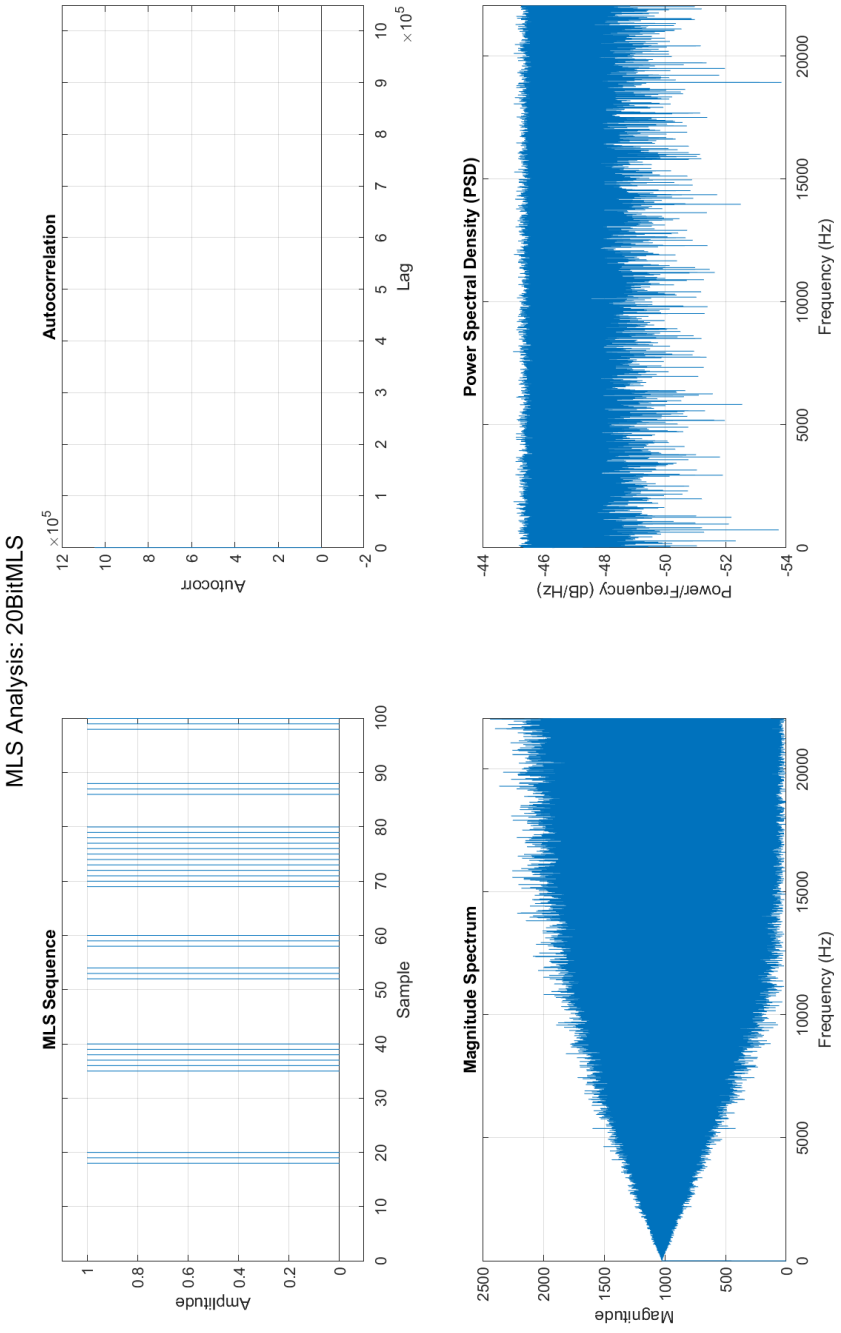


Figure I.18: Plots related to the 20-bit MLS sequence generated on the Teensy.

Sequence length: 2097151 samples
Detected register size: 21 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 2097151 samples
Register size (n) : 21 bits
Detected cyclic shift : 0
Autocorrelation peak : 2097151.000000 (ideal = 2097151) Max sidelobe deviation: 0.000000

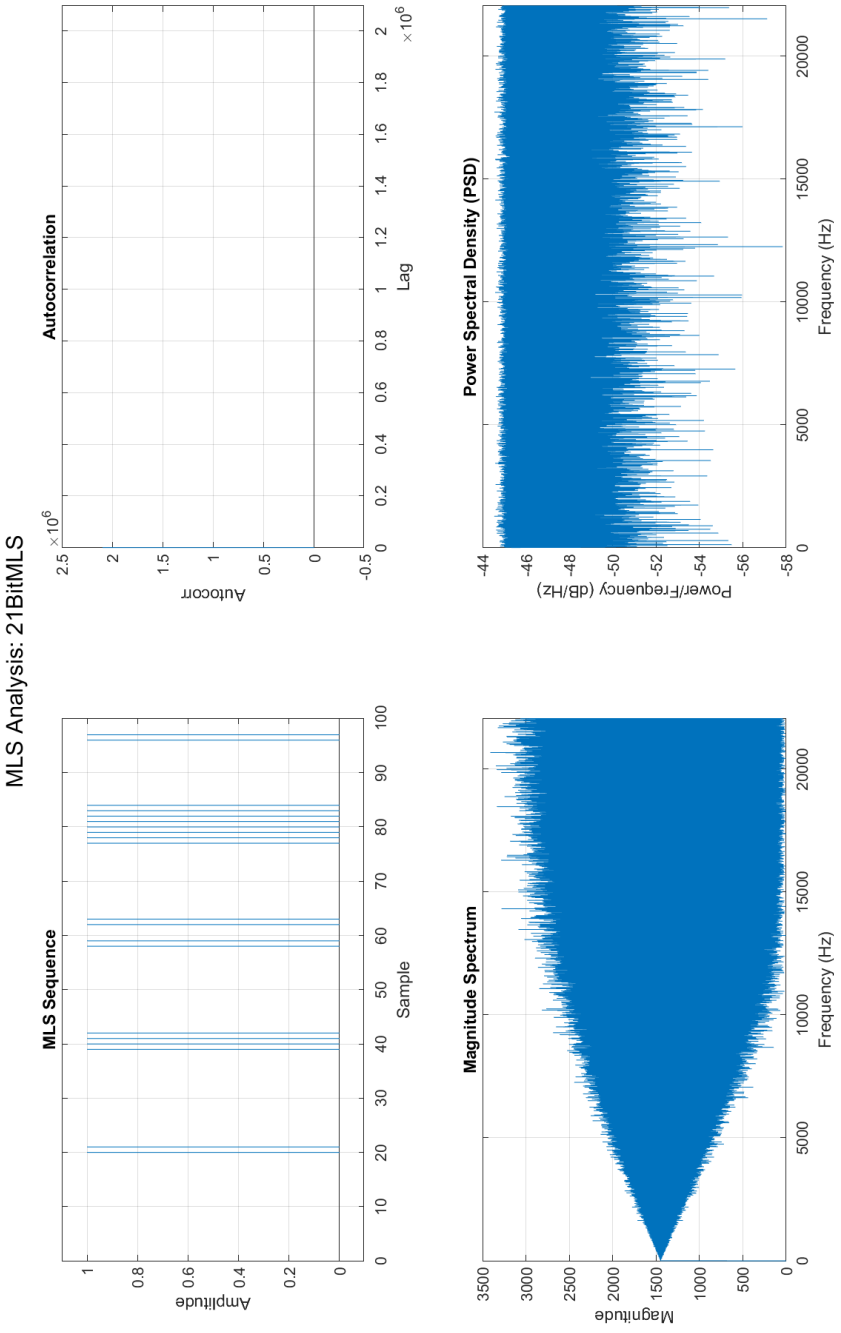


Figure I.19: Plots related to the 21-bit MLS sequence generated on the Teensy.

Sequence length: 4194303 samples
Detected register size: 22 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 4194303 samples
Register size (n) : 22 bits
Detected cyclic shift : 0
Autocorrelation peak : 4194303.000000 (ideal = 4194303)
Max sidelobe deviation: 0.000000

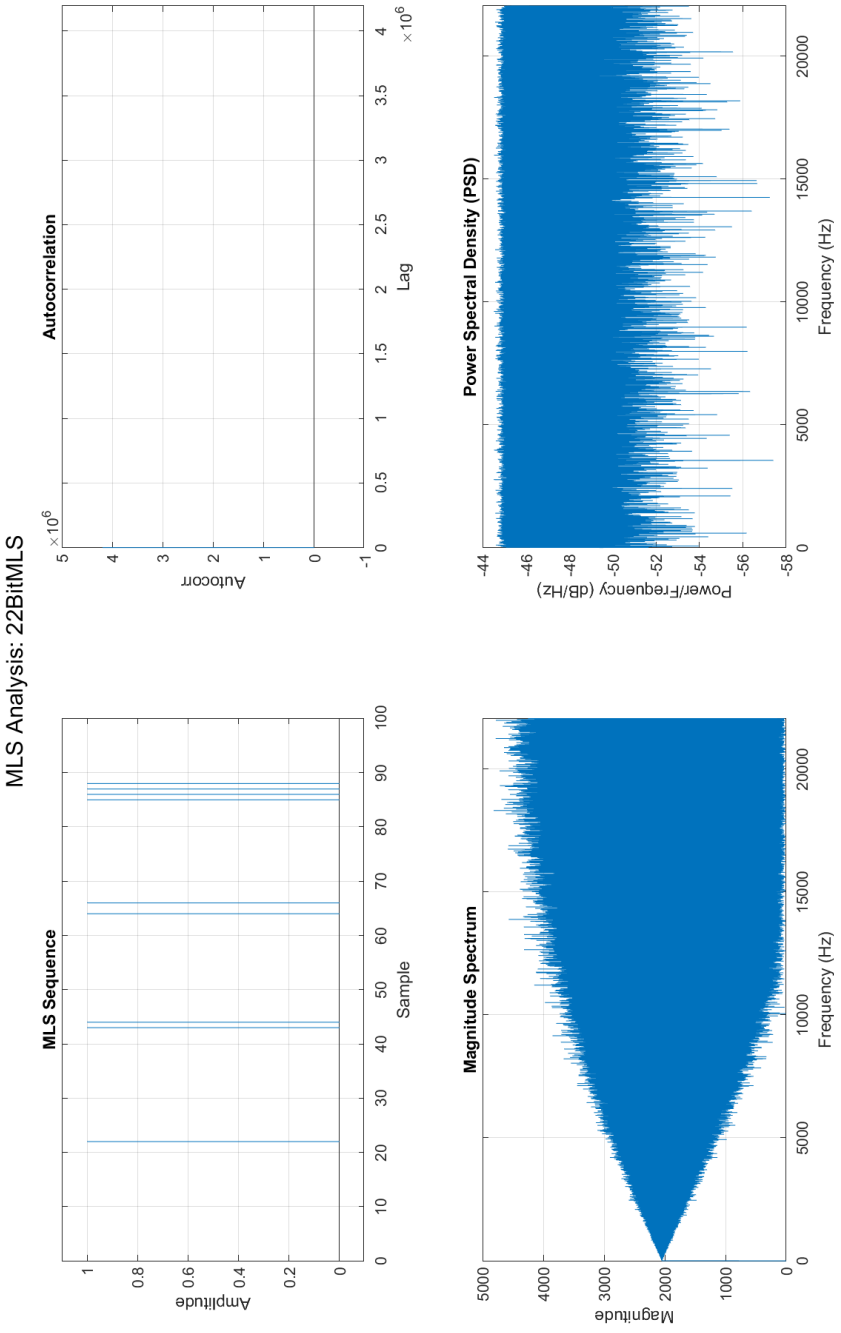


Figure I.20: Plots related to the 22-bit MLS sequence generated on the Teensy.

Sequence length: 8388607 samples
Detected register size: 23 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 8388607 samples
Register size (n) : 23 bits
Detected cyclic shift : 0
Autocorrelation peak : 8388607.000000 (ideal = 8388607)
Max sidelobe deviation: 0.000000

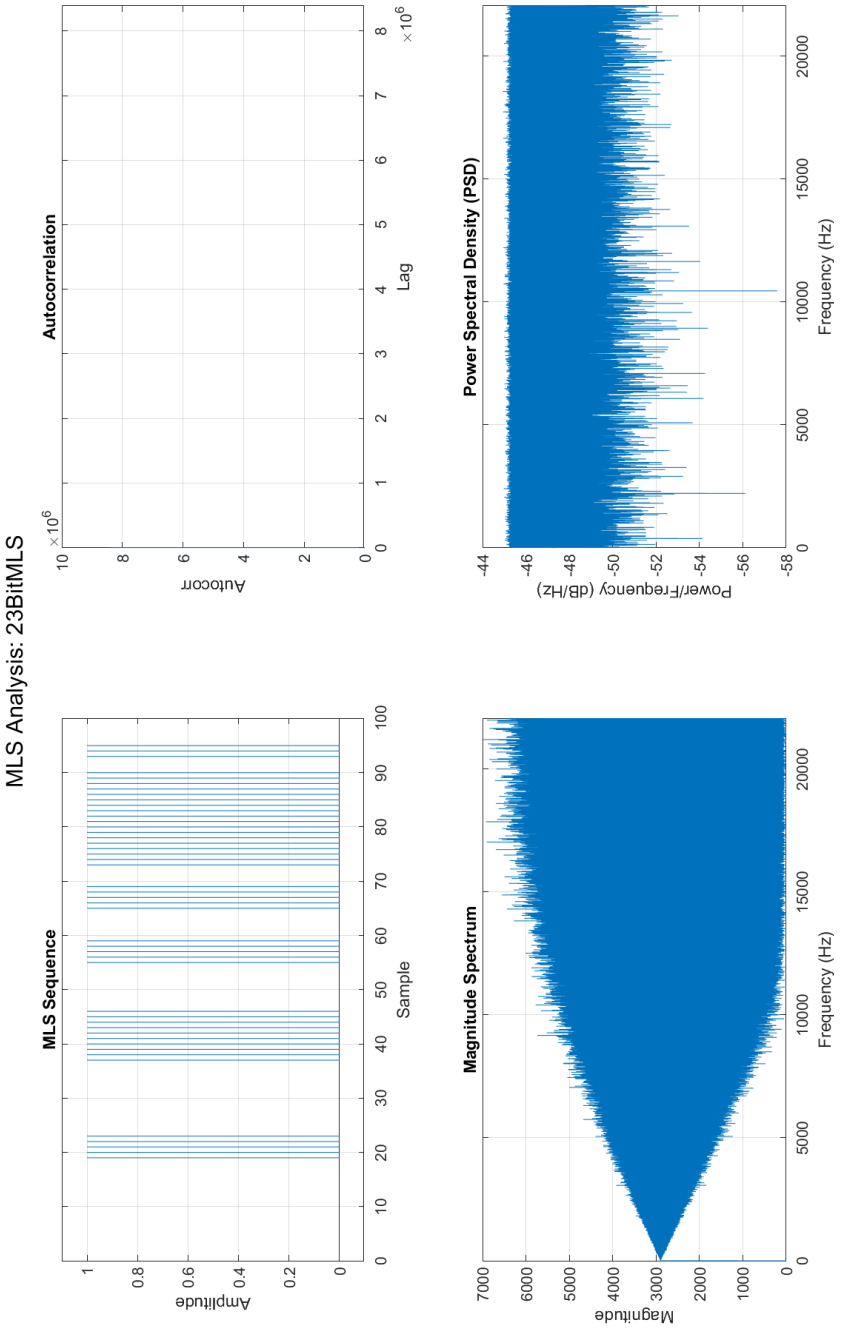


Figure I.21: Plots related to the 23-bit MLS sequence generated on the Teensy.

Sequence length: 16777215 samples
Detected register size: 24 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 16777215 samples
Register size (n) : 24 bits
Detected cyclic shift : 0
Autocorrelation peak : 16777215.000000 (ideal = 16777215)
Max sidelobe deviation: 0.000000

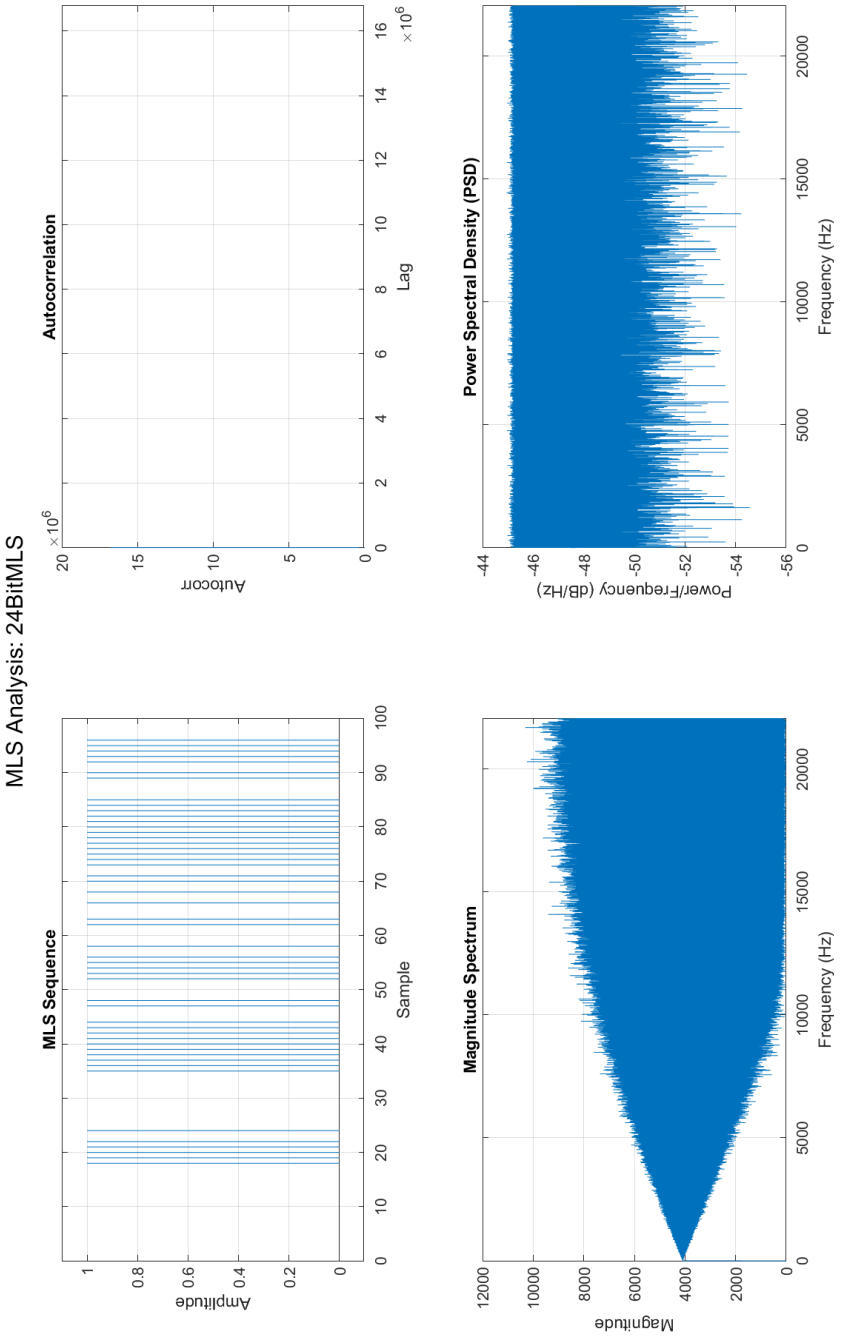


Figure I.22: Plots related to the 24-bit MLS sequence generated on the Teensy.

Sequence length: 33554431 samples
Detected register size: 25 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 33554431 samples
Register size (n) : 25 bits
Detected cyclic shift : 0
Autocorrelation peak : 33554431.000000 (ideal = 33554431)
Max sidelobe deviation: 0.000000

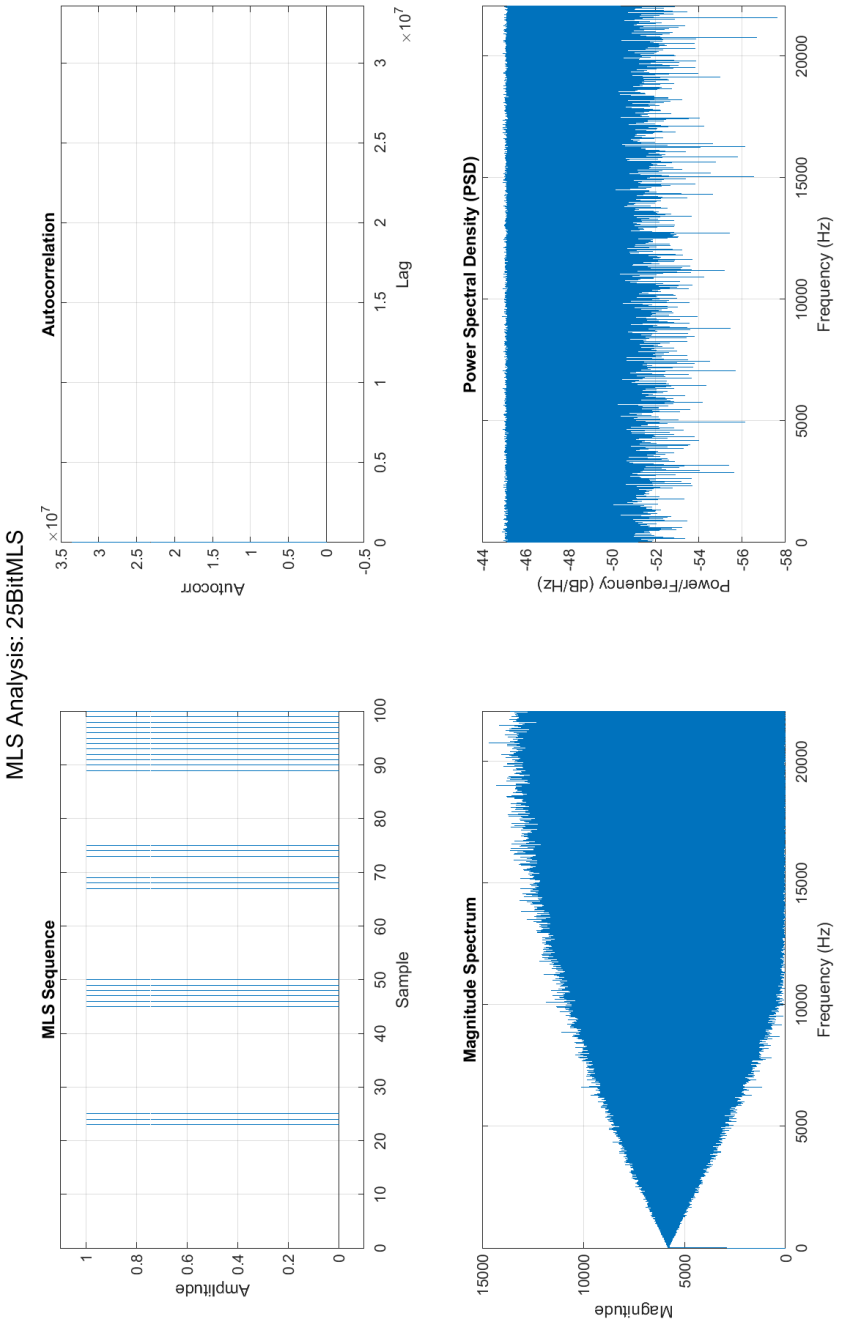


Figure I.23: Plots related to the 25-bit MLS sequence generated on the Teensy.

Sequence length: 67108863 samples
Detected register size: 26 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 67108863 samples
Register size (n) : 26 bits
Detected cyclic shift : 0
Autocorrelation peak : 67108863.000000 (ideal = 67108863)
Max sidelobe deviation: 0.000000

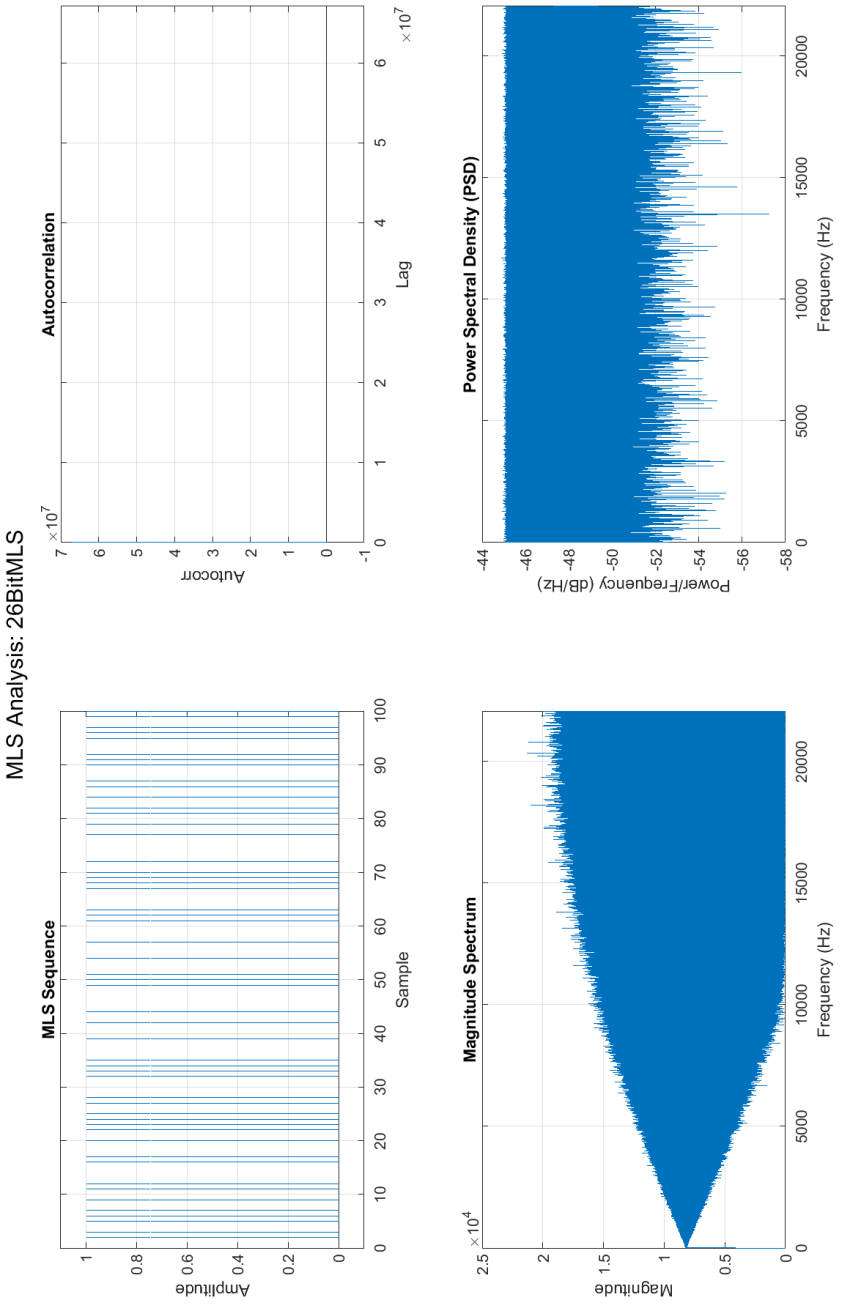


Figure I.24: Plots related to the 26-bit MLS sequence generated on the Teensy.

Sequence length: 134217727 samples
Detected register size: 27 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 134217727 samples
Register size (n) : 27 bits
Detected cyclic shift : 0
Autocorrelation peak : 134217727.000000 (ideal = 134217727)
Max sidelobe deviation: 0.000000

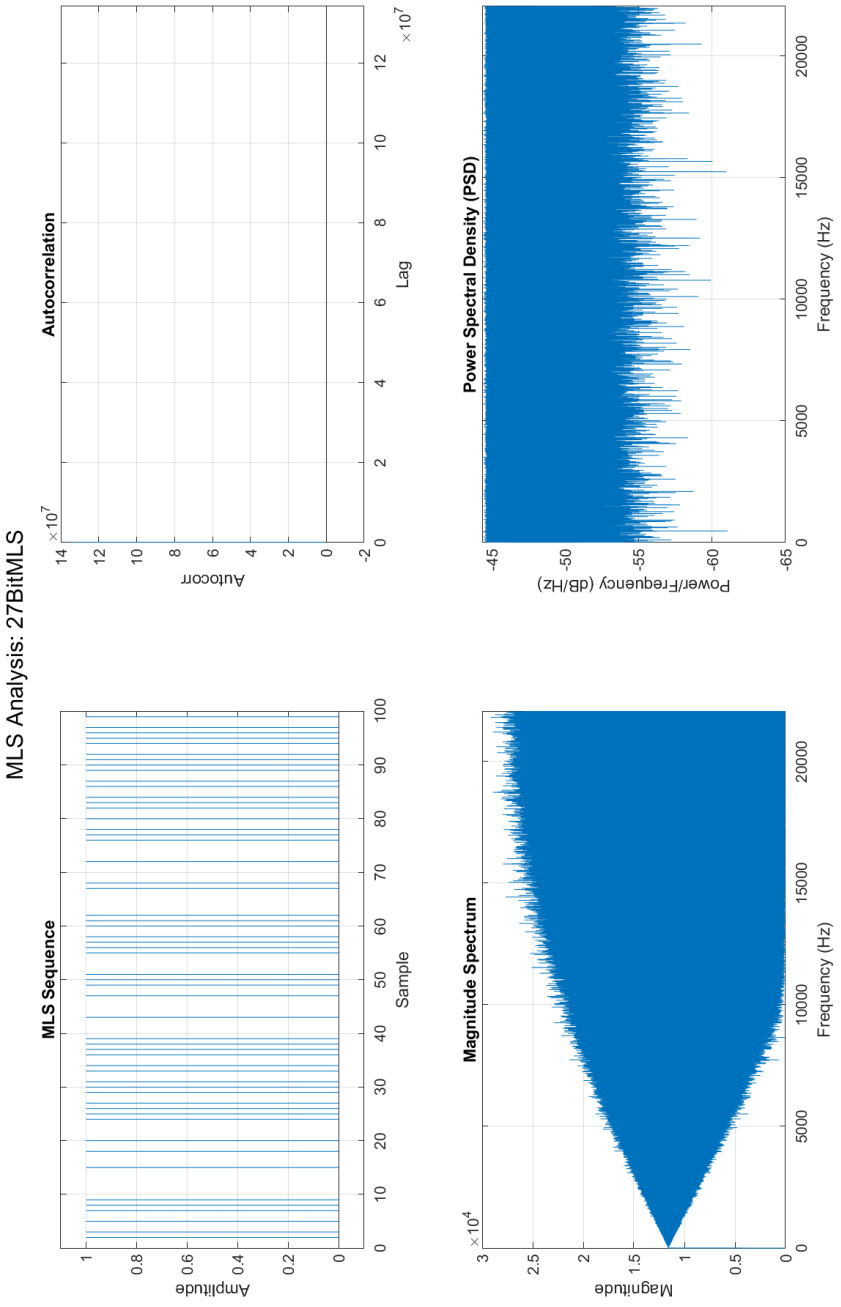


Figure I.25: Plots related to the 27-bit MLS sequence generated on the Teensy.

Sequence length: 268435456 samples
Detected register size: 28 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 268435456 samples
Register size (n) : 28 bits
Detected cyclic shift : 0
Autocorrelation peak : 268435456.000000 (ideal = 268435456)
Max sidelobe deviation: 0.000000

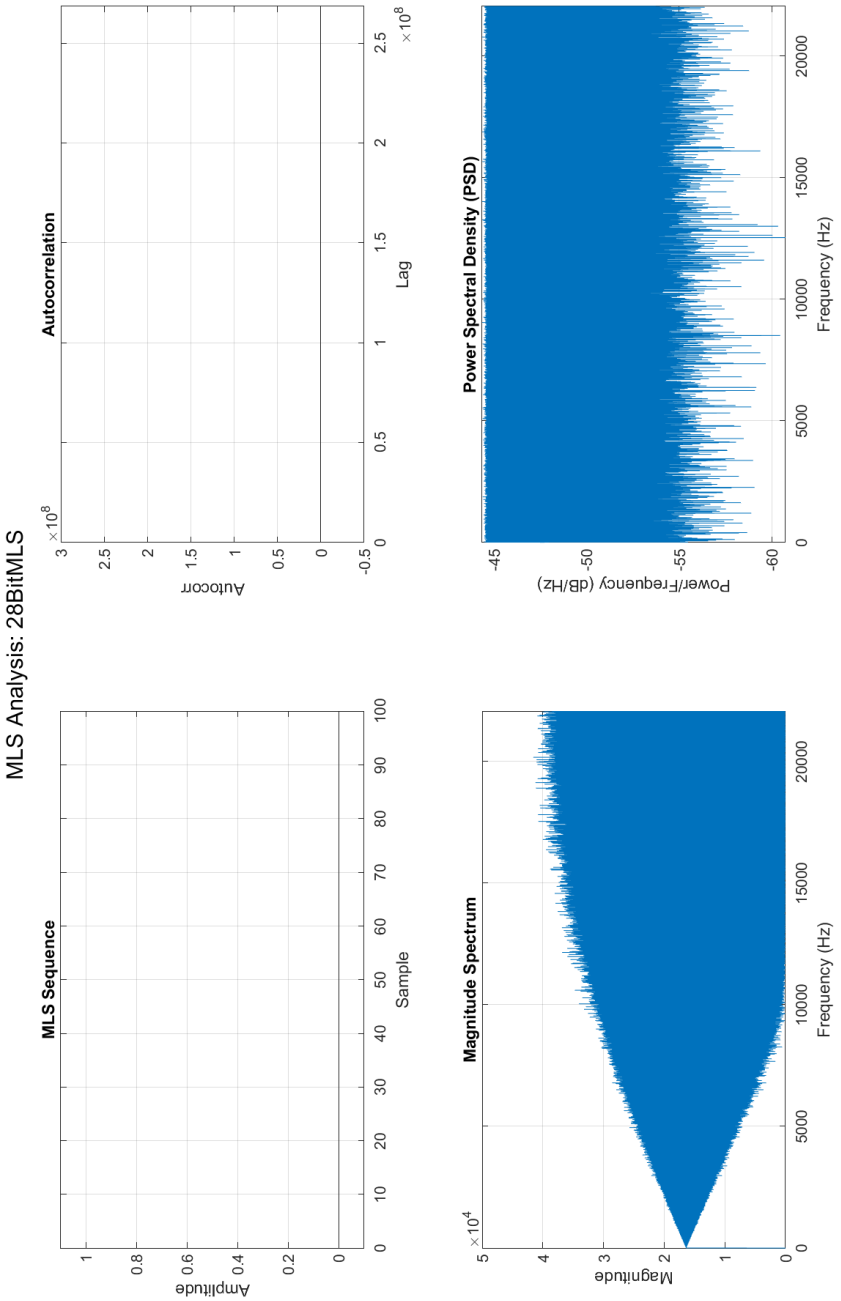


Figure I.26: Plots related to the 28-bit MLS sequence generated on the Teensy.

Sequence length: 536870911 samples
Detected register size: 29 bits
✔Sequence is a VALID MLS.
Detected cyclic shift of 0 samples.
Summary:
Length : 536870911 samples
Register size (n) : 29 bits
Detected cyclic shift : 0
Autocorrelation peak : 536870911.000000 (ideal = 536870911)
Max sidelobe deviation: 0.000000

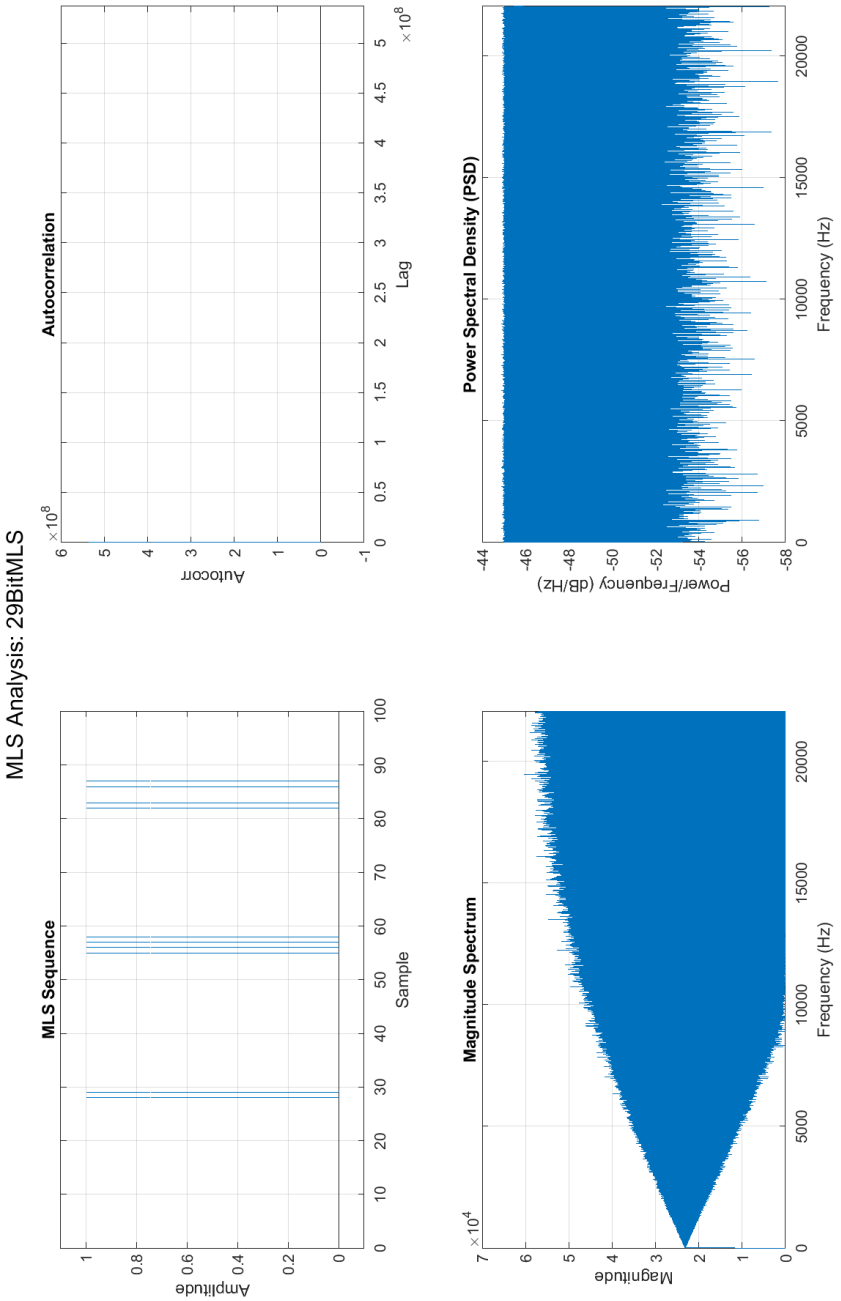


Figure I.27: Plots related to the 29-bit MLS sequence generated on the Teensy.

I.3 Faulty Teensy Generated MLS - MLSChecker Results and Plots

Sequence length: 15 samples
Detected register size: 4 bits
✖Sequence FAILED MLS check.
Length : 15 samples
Register size (n) : 4 bits
Detected cyclic shift : 0
Autocorrelation peak : 15.000000 (ideal = 15)
Max sidelobe deviation: 4.000000

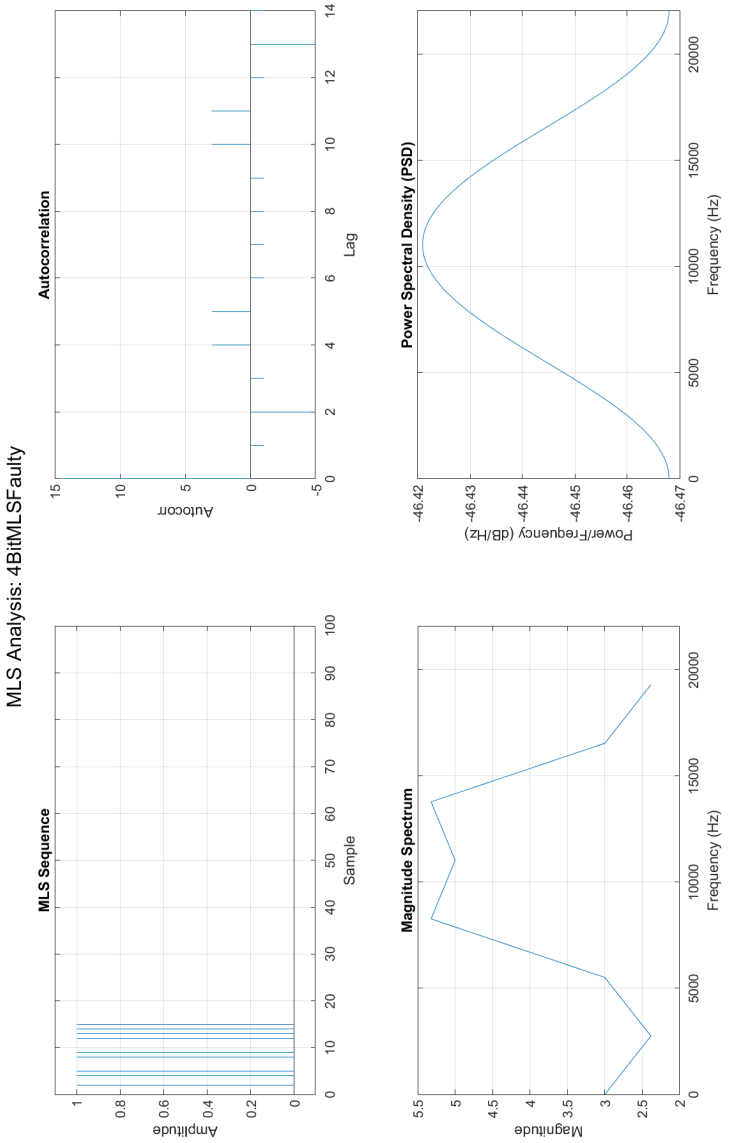


Figure I.28: Plots related to the faulty 4-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 31 samples
Detected register size: 5 bits
✖Sequence FAILED MLS check.
Length : 31 samples
Register size (n) : 5 bits
Detected cyclic shift : 0
Autocorrelation peak : 31.000000 (ideal = 31) Max sidelobe deviation: 4.000000

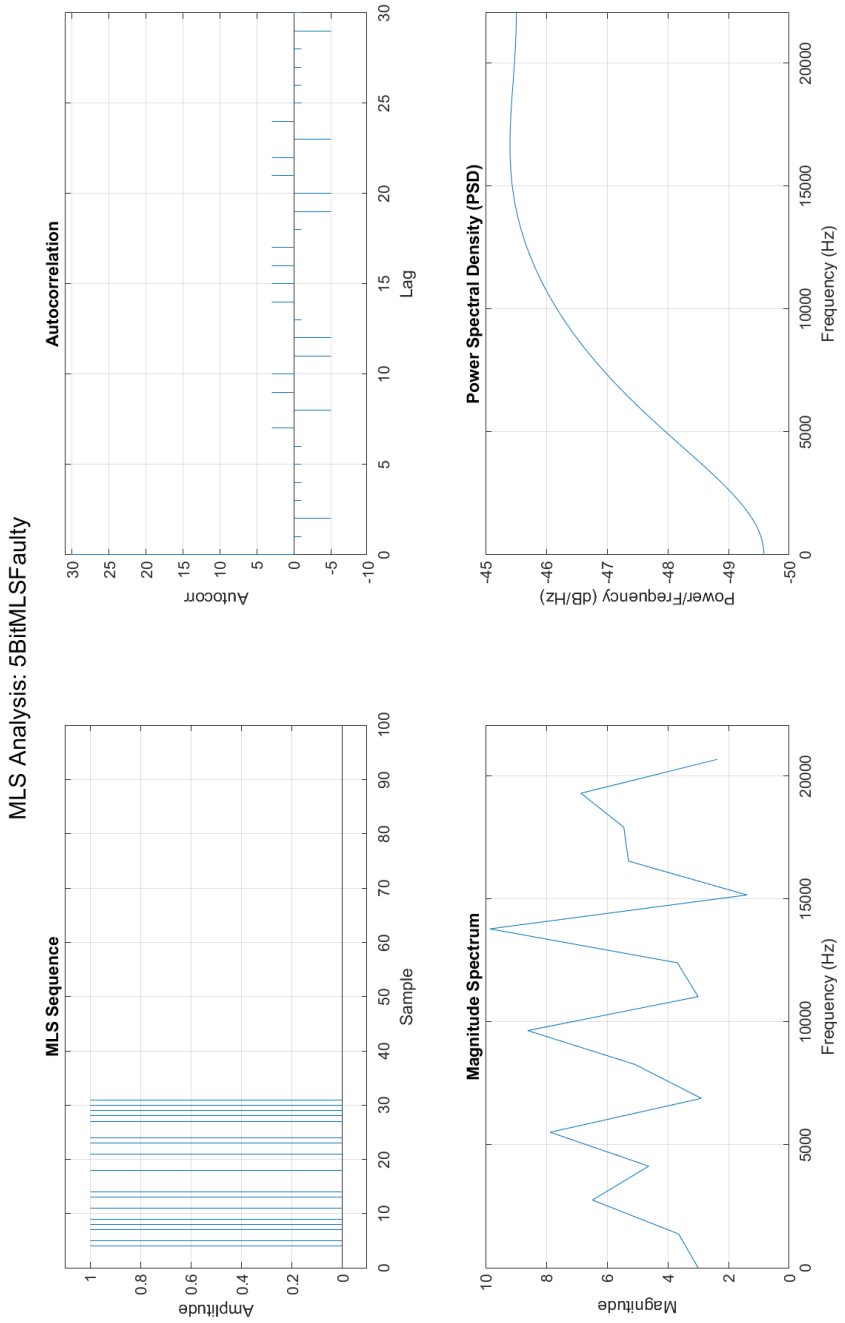


Figure I.29: Plots related to the faulty 5-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 63 samples
Detected register size: 6 bits
✖Sequence FAILED MLS check.
Length : 63 samples
Register size (n) : 6 bits
Detected cyclic shift : 0
Autocorrelation peak : 63.000000 (ideal = 63)
Max sidelobe deviation: 4.000000

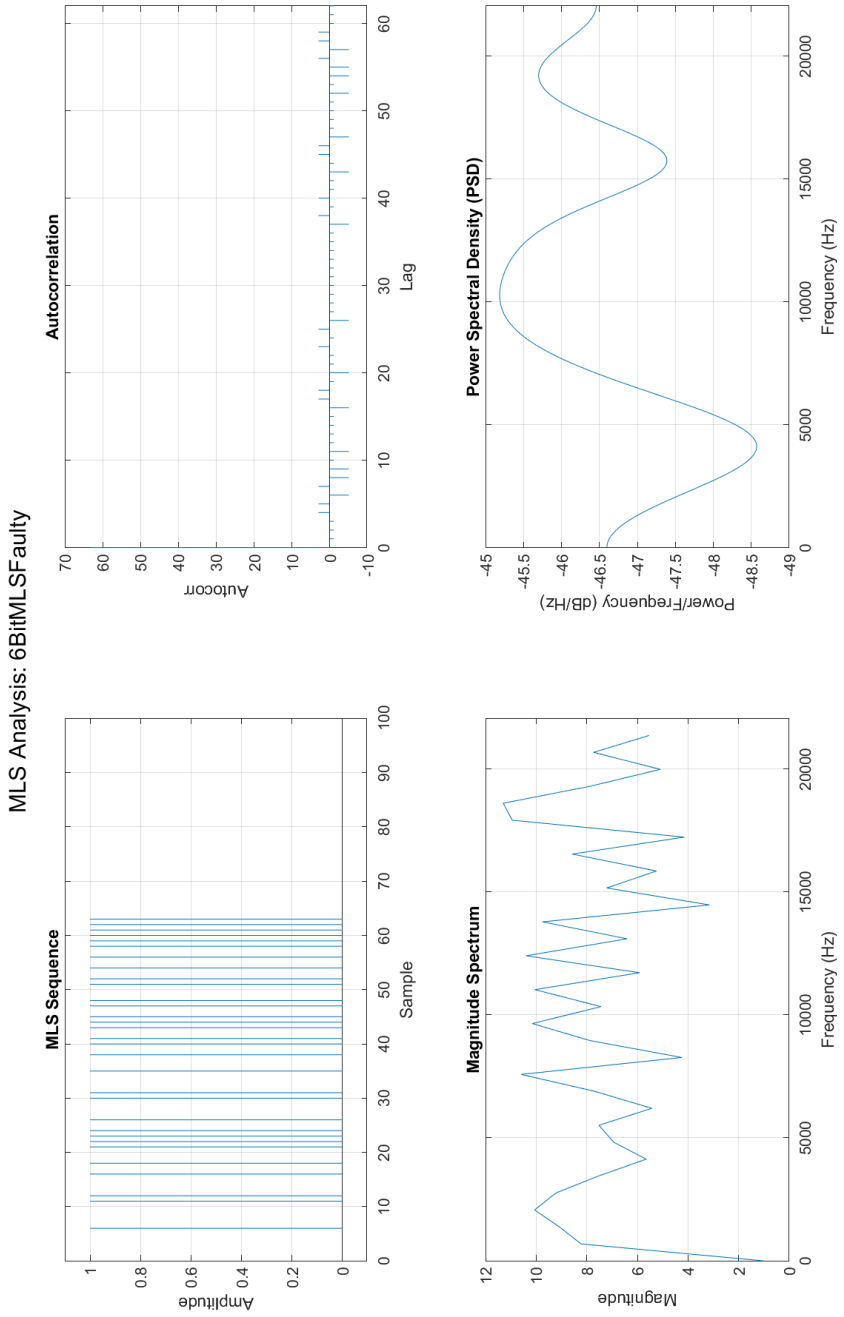


Figure I.30: Plots related to the faulty 6-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 127 samples
Detected register size: 7 bits
✖Sequence FAILED MLS check.
Length : 127 samples
Register size (n) : 7 bits
Detected cyclic shift : 0
Autocorrelation peak : 127.000000 (ideal = 127)
Max sidelobe deviation: 4.000000

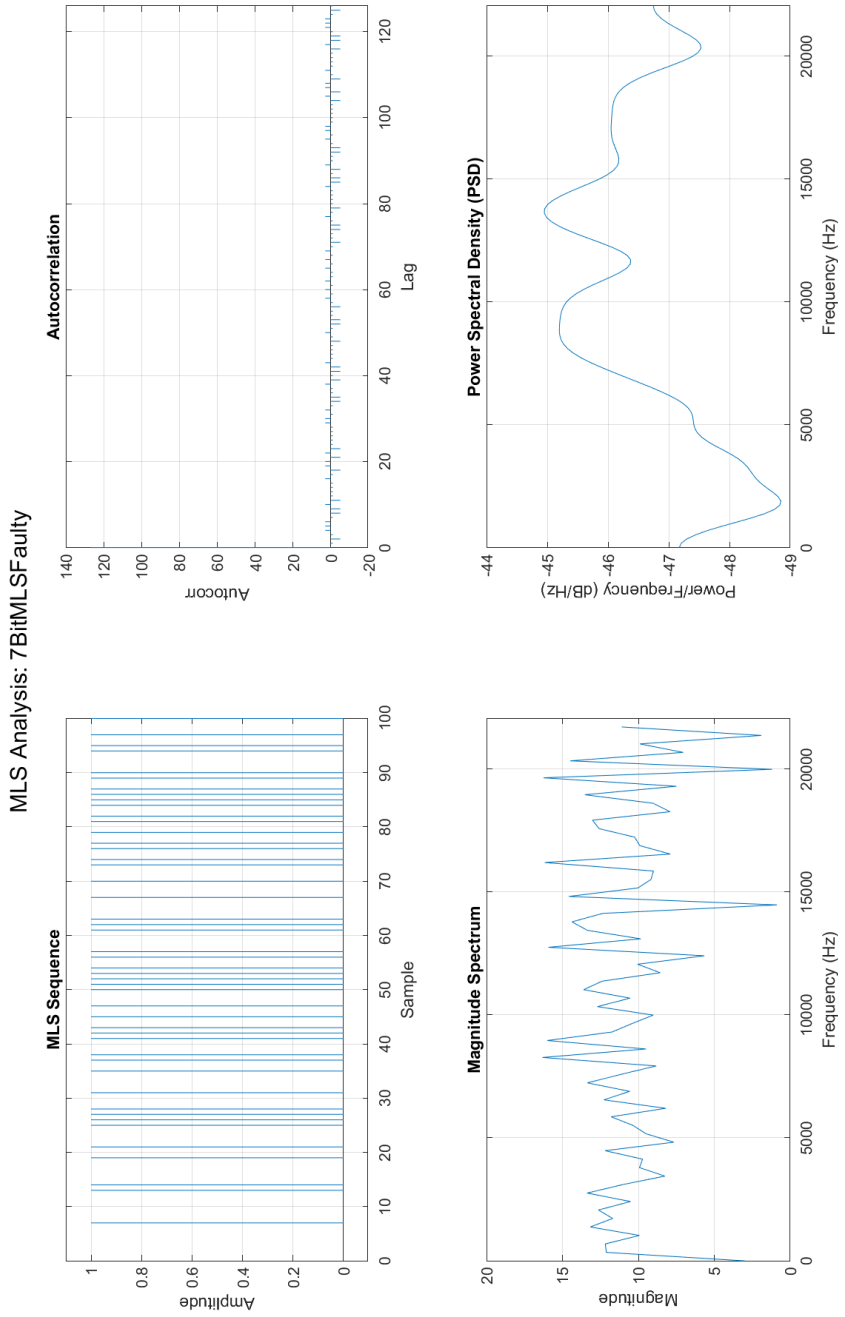


Figure I.31: Plots related to the faulty 7-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 255 samples
Detected register size: 8 bits
✖Sequence FAILED MLS check.
Length : 255 samples
Register size (n) : 8 bits
Detected cyclic shift : 0
Autocorrelation peak : 255.000000 (ideal = 255)
Max sidelobe deviation: 4.000000

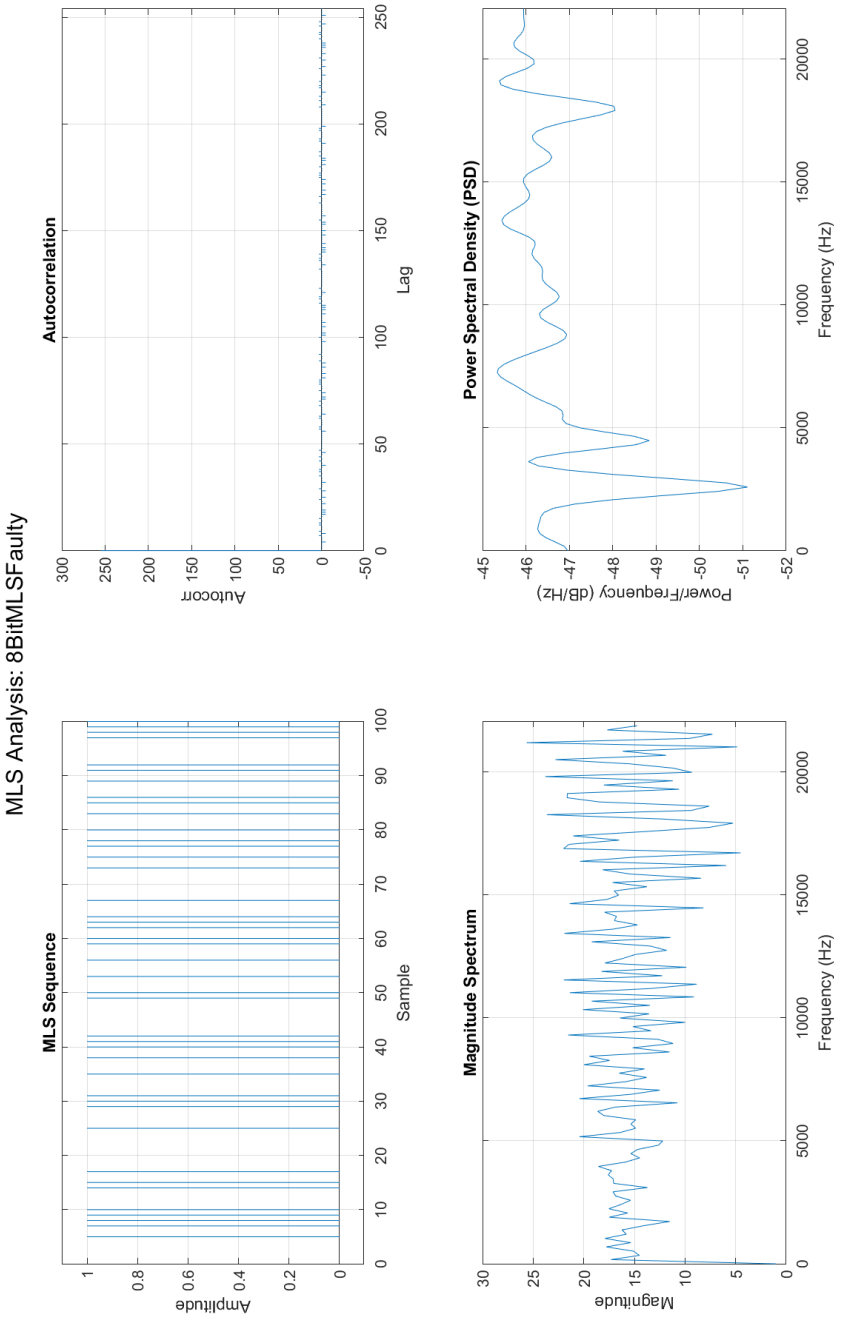


Figure I.32: Plots related to the faulty 8-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 511 samples
Detected register size: 9 bits
✖Sequence FAILED MLS check.
Length : 511 samples
Register size (n) : 9 bits
Detected cyclic shift : 0
Autocorrelation peak : 511.000000 (ideal = 511) Max sidelobe deviation: 4.000000

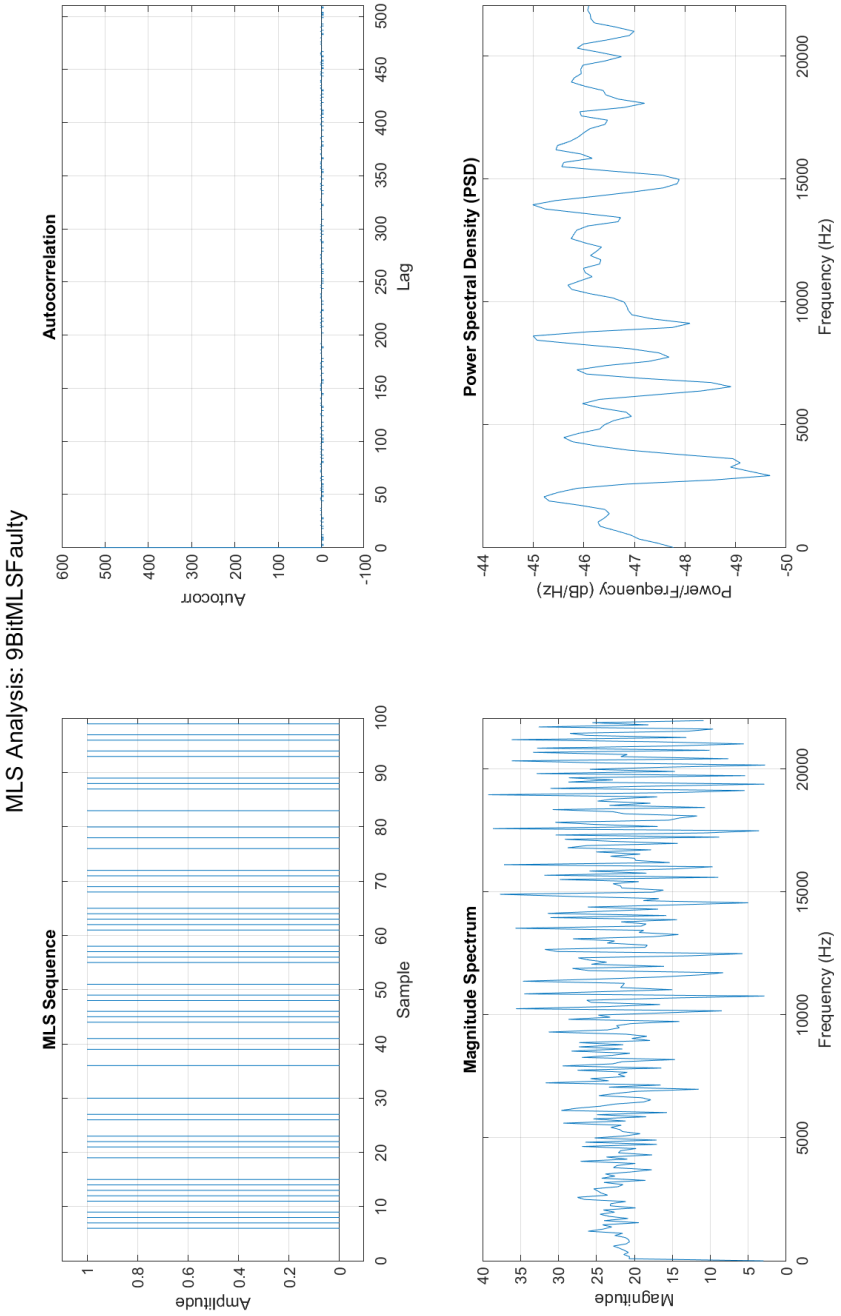


Figure I.33: Plots related to the faulty 9-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 1023 samples
Detected register size: 10 bits
✖Sequence FAILED MLS check.
Length : 1023 samples
Register size (n) : 10 bits
Detected cyclic shift : 0
Autocorrelation peak : 1023.000000 (ideal = 1023)
Max sidelobe deviation: 4.000000

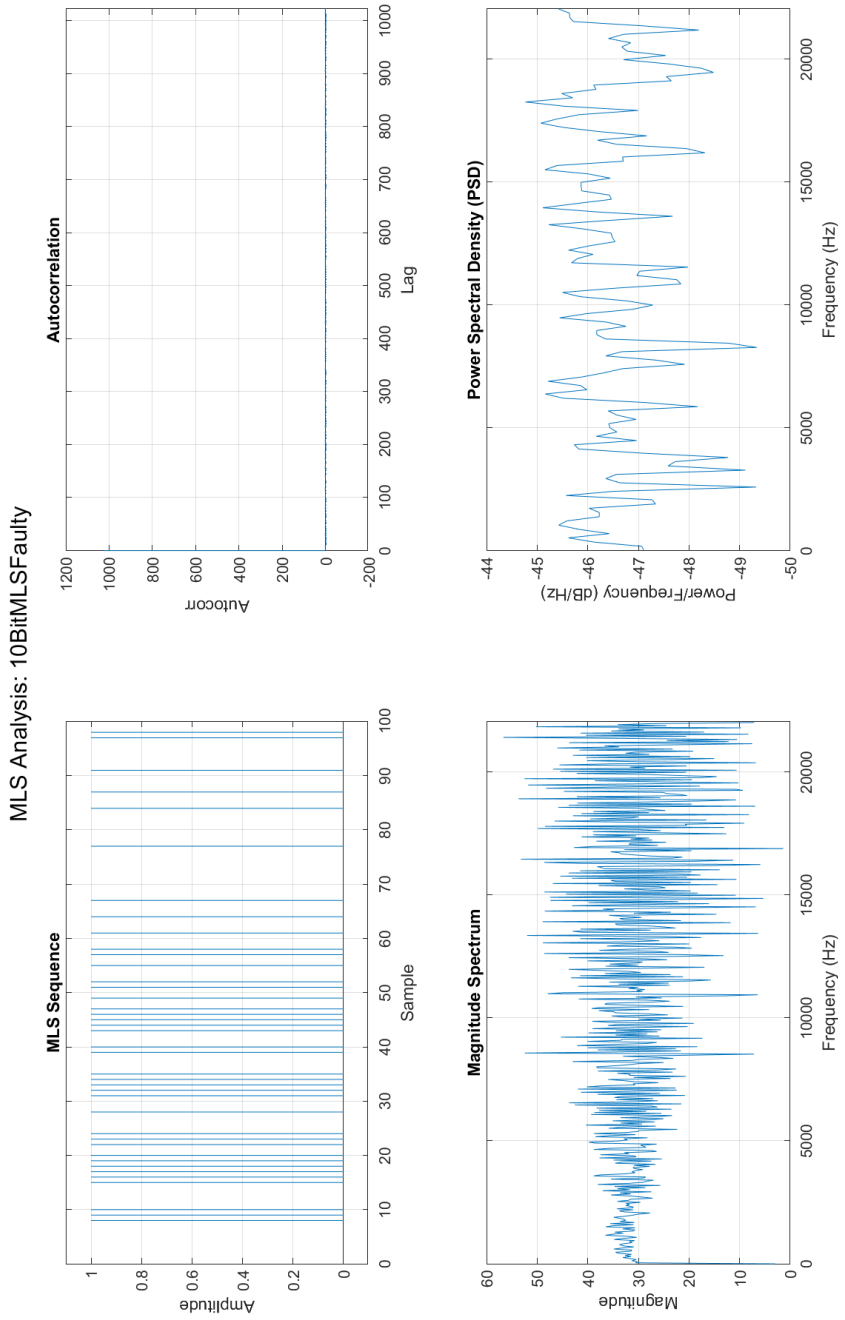


Figure I.34: Plots related to the faulty 10-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 2047 samples
Detected register size: 11 bits
✖Sequence FAILED MLS check.
Length : 2047 samples
Register size (n) : 11 bits
Detected cyclic shift : 0
Autocorrelation peak : 2047.000000 (ideal = 2047)
Max sidelobe deviation: 4.000000

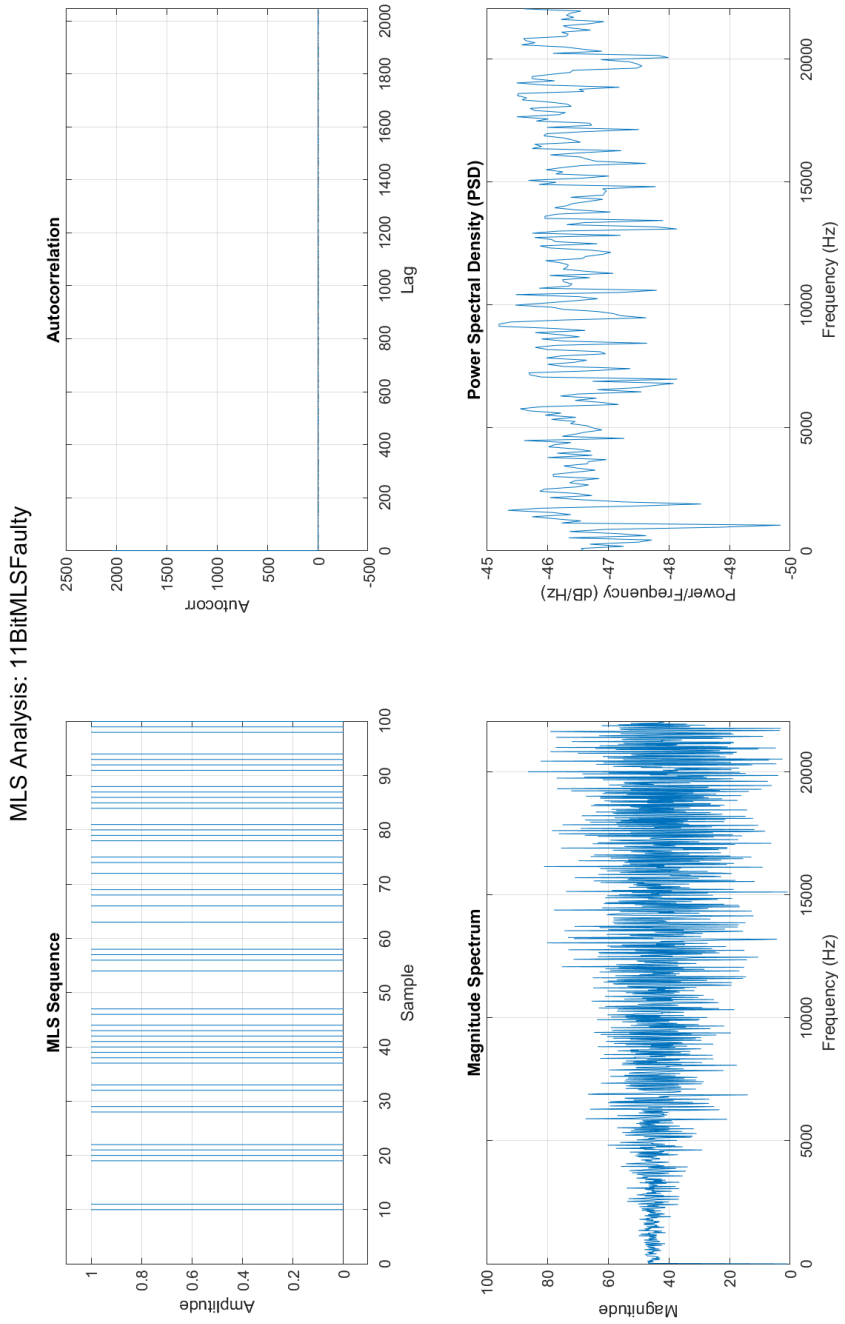


Figure I.35: Plots related to the faulty 11-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 4095 samples
Detected register size: 12 bits
✖Sequence FAILED MLS check.
Length : 4095 samples
Register size (n) : 12 bits
Detected cyclic shift : 0
Autocorrelation peak : 4095.000000 (ideal = 4095)
Max sidelobe deviation: 4.000000

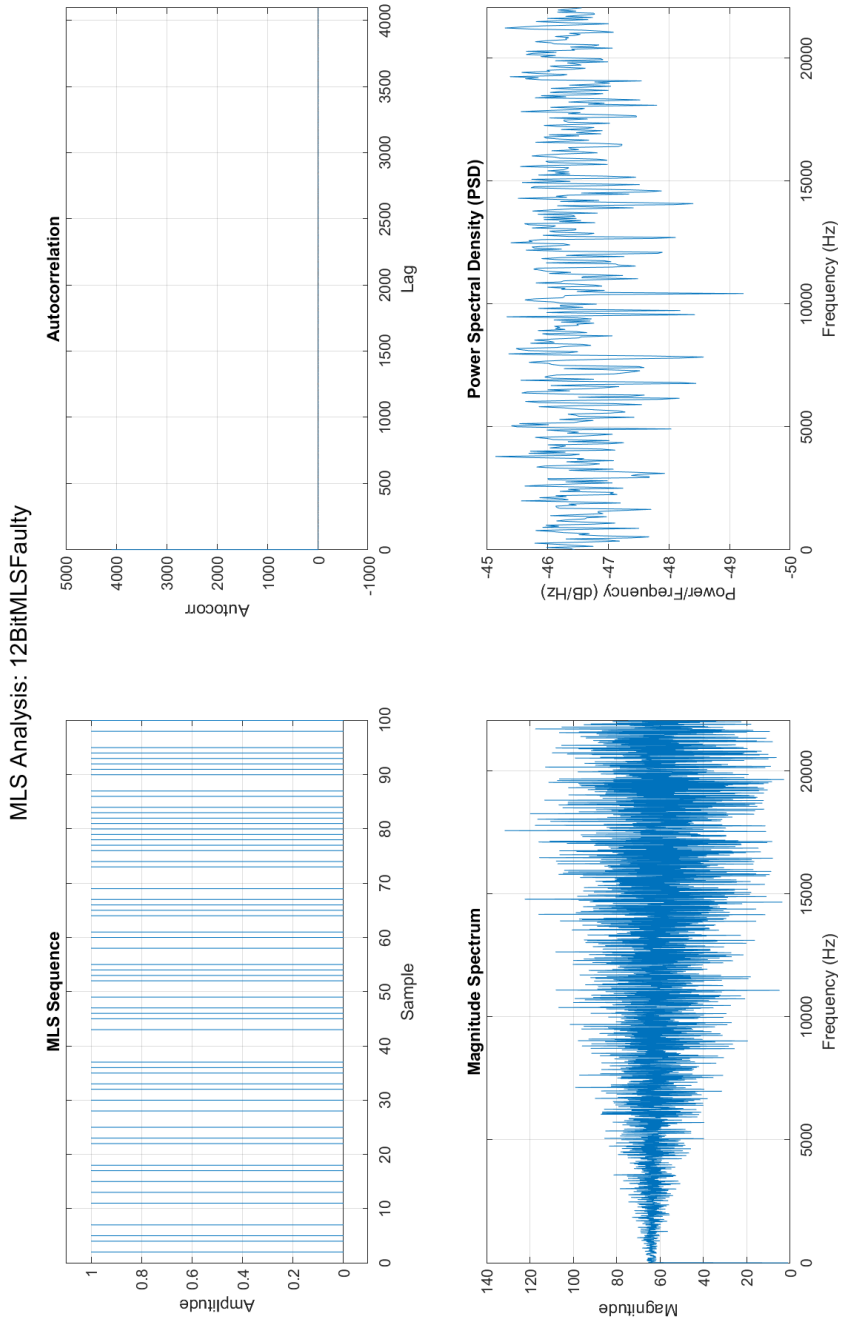


Figure I.36: Plots related to the faulty 12-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 8191 samples
Detected register size: 13 bits
✖Sequence FAILED MLS check.
Length : 8191 samples
Register size (n) : 13 bits
Detected cyclic shift : 0
Autocorrelation peak : 8191.000000 (ideal = 8191) Max sidelobe deviation: 4.000000

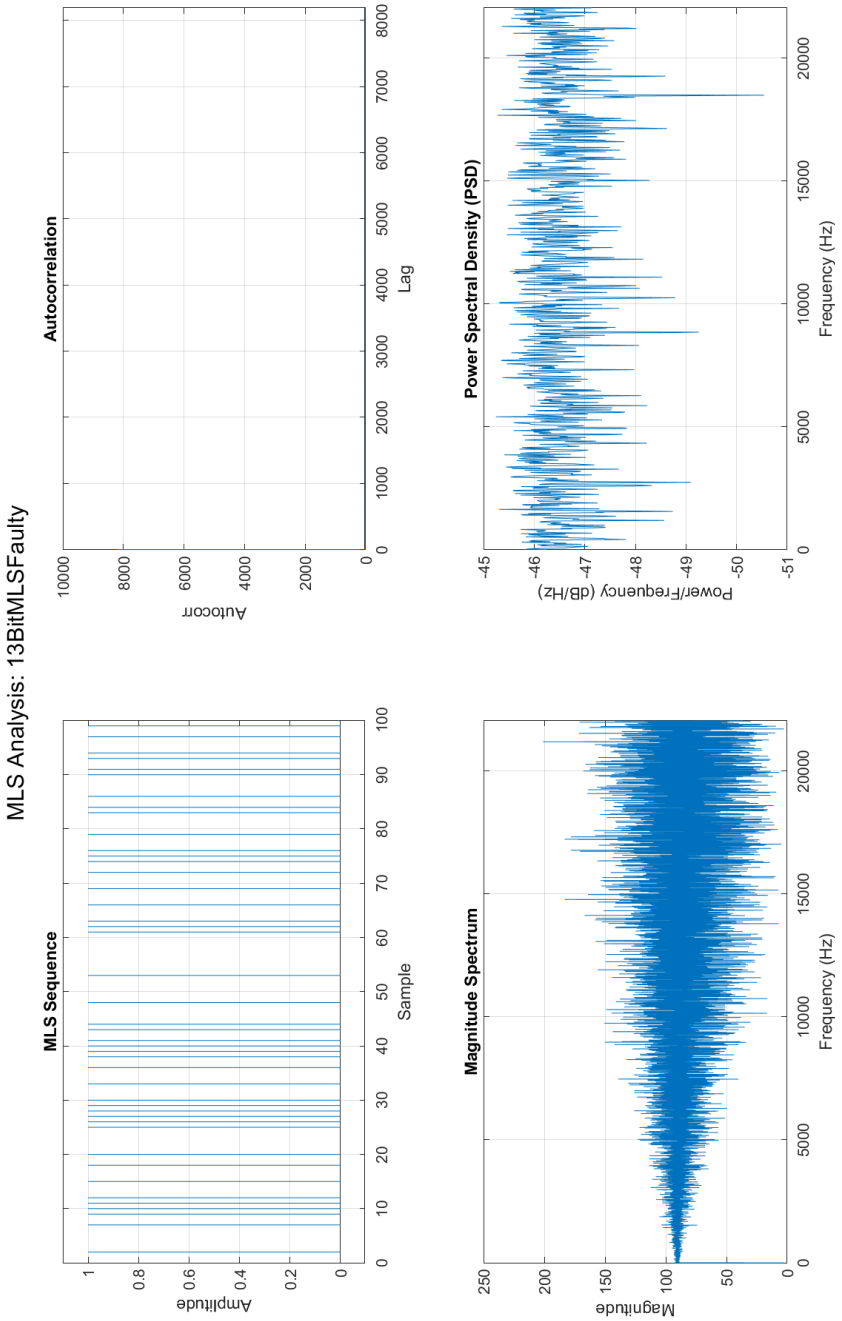


Figure I.37: Plots related to the faulty 13-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 16383 samples
Detected register size: 14 bits
✖Sequence FAILED MLS check.
Length : 16383 samples
Register size (n) : 14 bits
Detected cyclic shift : 0
Autocorrelation peak : 16383.000000 (ideal = 16383)
Max sidelobe deviation: 4.000000

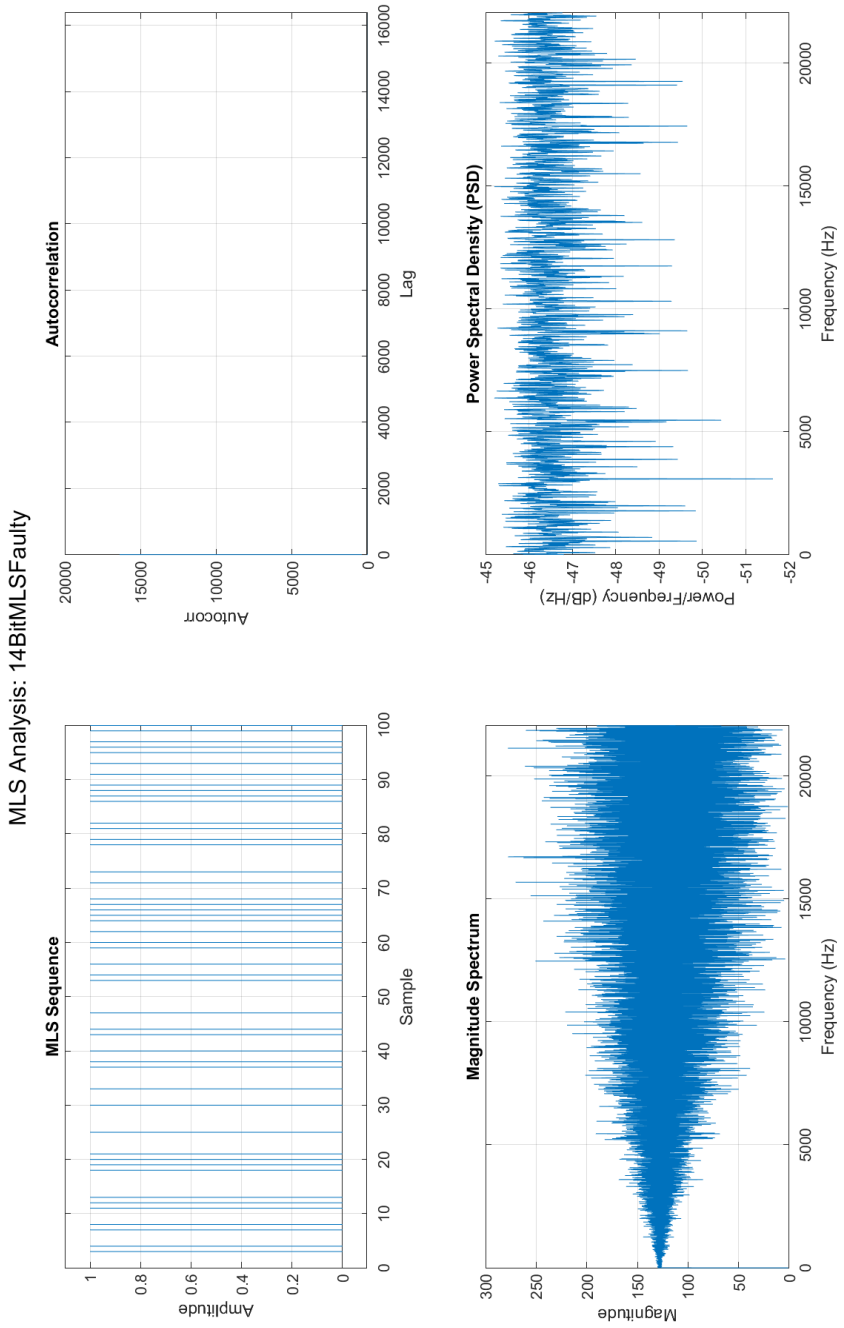


Figure I.38: Plots related to the faulty 14-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 32767 samples
Detected register size: 15 bits
✖Sequence FAILED MLS check.
Length : 32767 samples
Register size (n) : 15 bits
Detected cyclic shift : 0
Autocorrelation peak : 32767.000000 (ideal = 32767)
Max sidelobe deviation: 4.000000

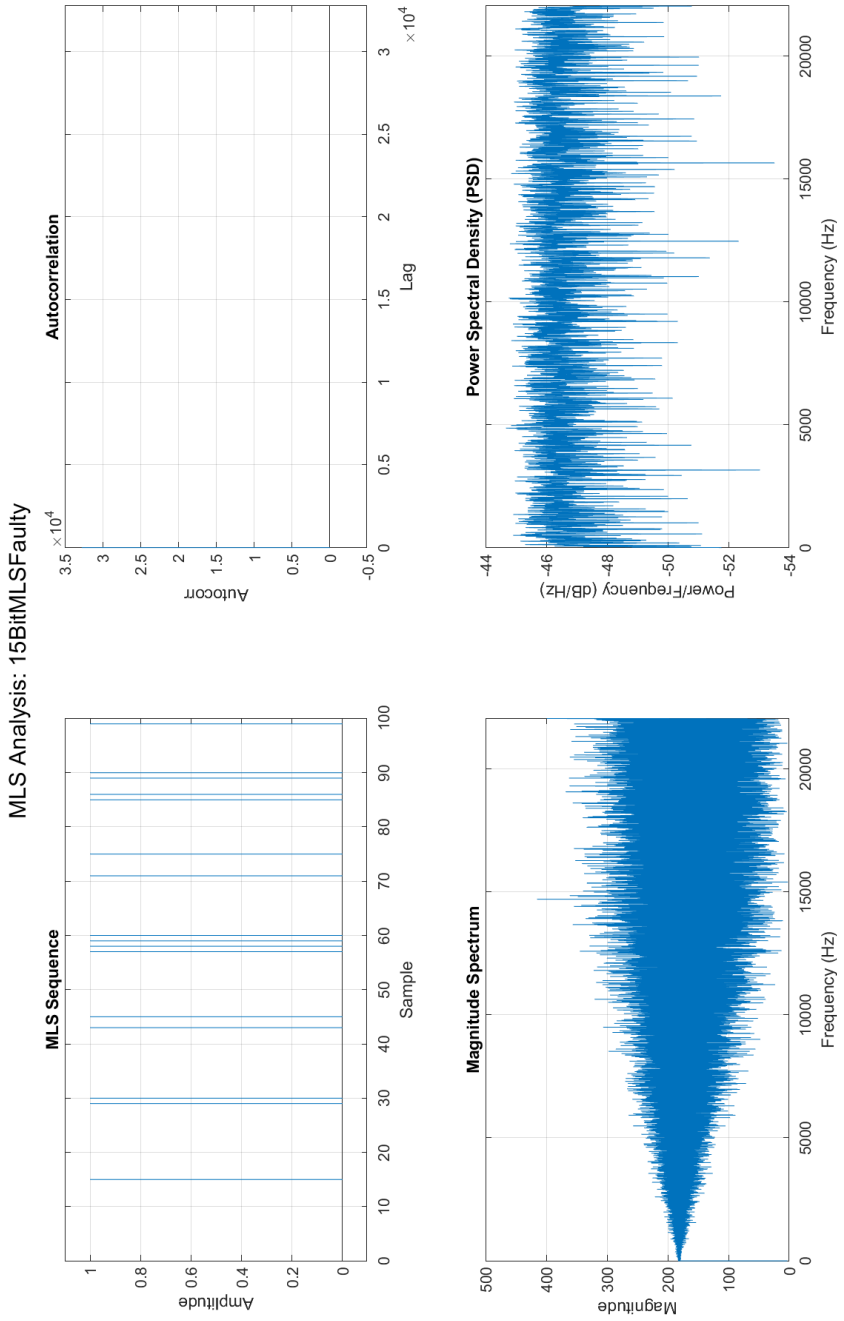


Figure I.39: Plots related to the faulty 15-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 65535 samples
Detected register size: 16 bits
✖Sequence FAILED MLS check.
Length : 65535 samples
Register size (n) : 16 bits
Detected cyclic shift : 0
Autocorrelation peak : 65535.000000 (ideal = 65535)
Max sidelobe deviation: 4.000000

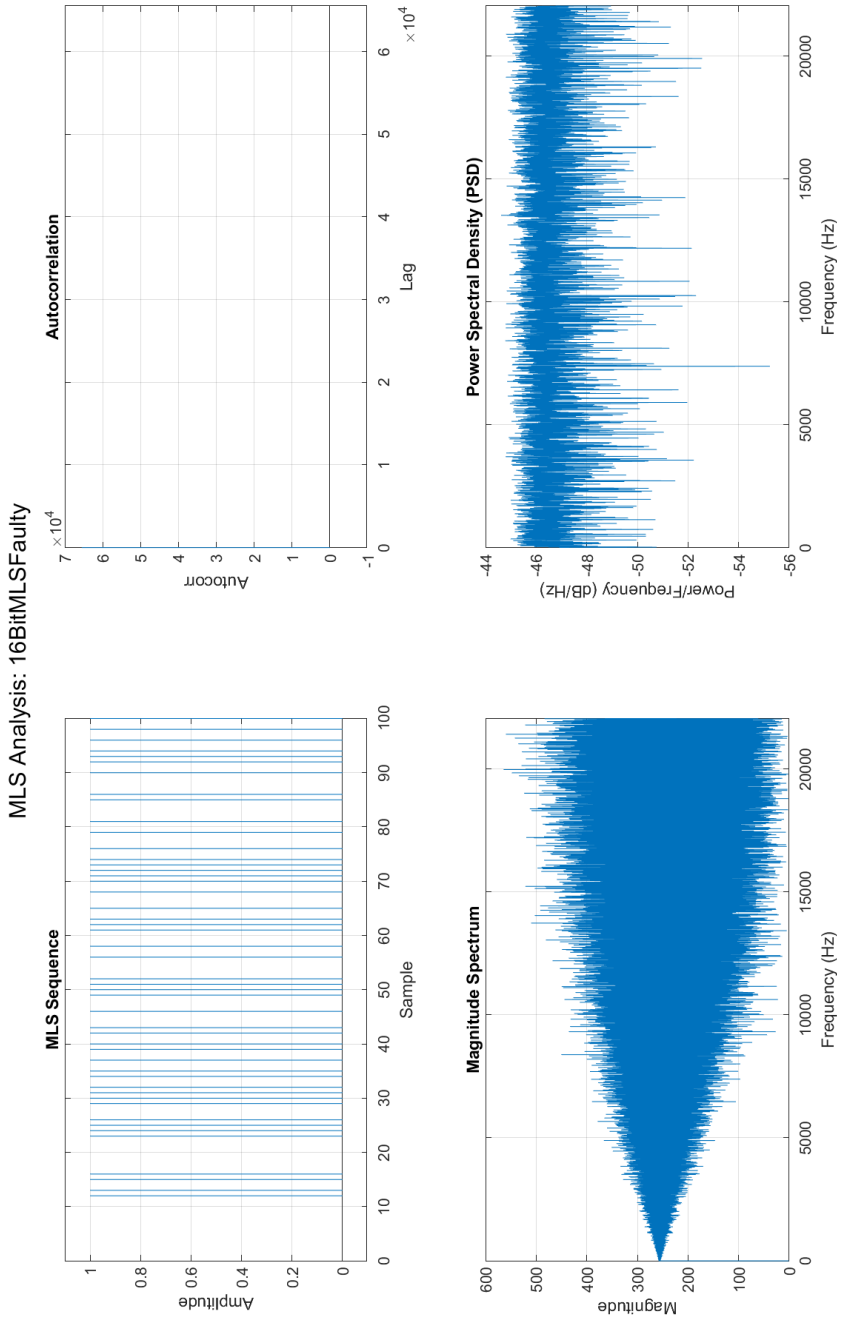


Figure I.40: Plots related to the faulty 16-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 131071 samples
Detected register size: 17 bits
✖Sequence

Length : 131071 samples
Register size (n) : 17 bits
Detected cyclic shift : NaN
Autocorrelation peak : 131071.000000 (ideal = 131071) Max sidelobe deviation: 4.000000

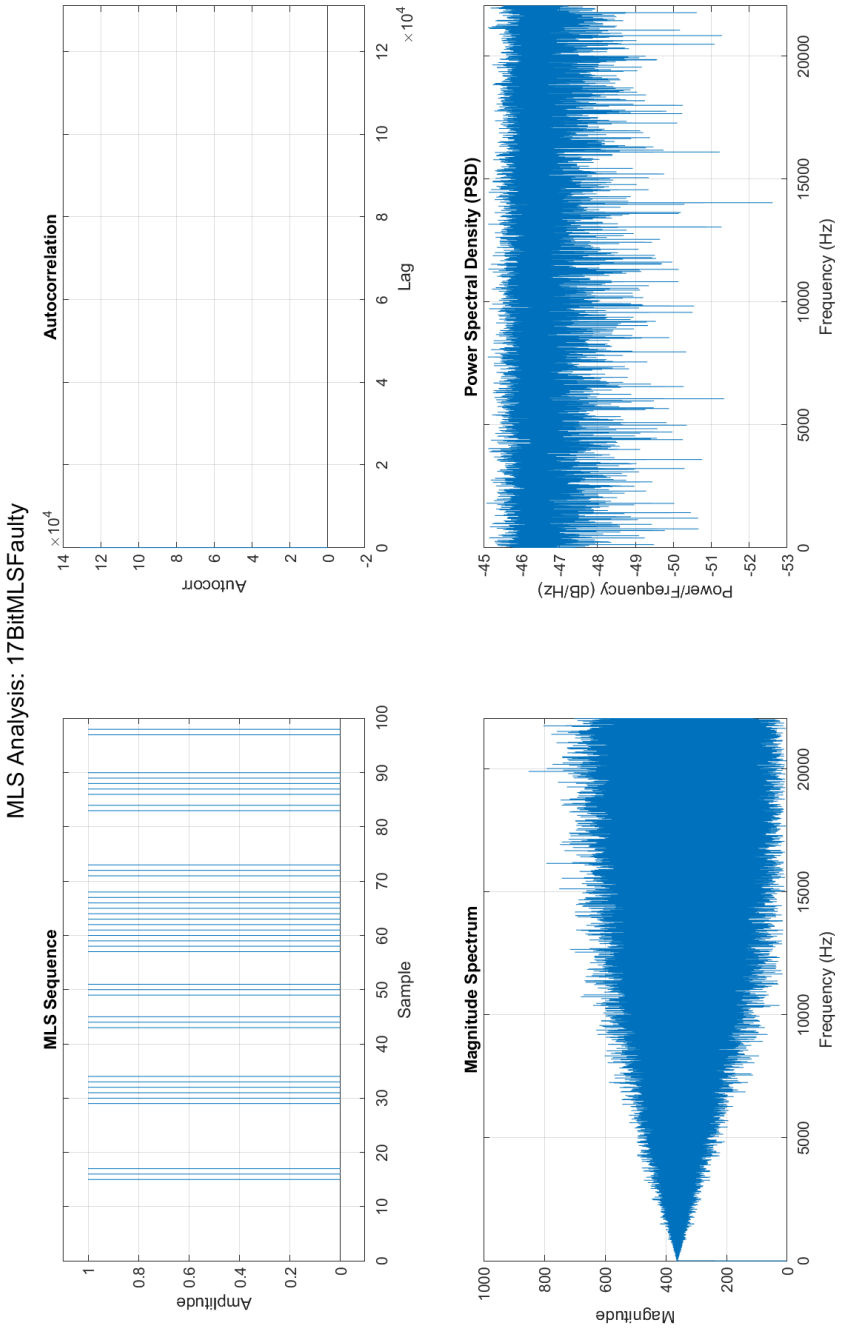


Figure I.41: Plots related to the faulty 17-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 262143 samples
Detected register size: 18 bits
✖Sequence FAILED MLS check.
Length : 262143 samples
Register size (n) : 18 bits
Detected cyclic shift : NaN
Autocorrelation peak : 262143.000000 (ideal = 262143)
Max sidelobe deviation: 4.000000

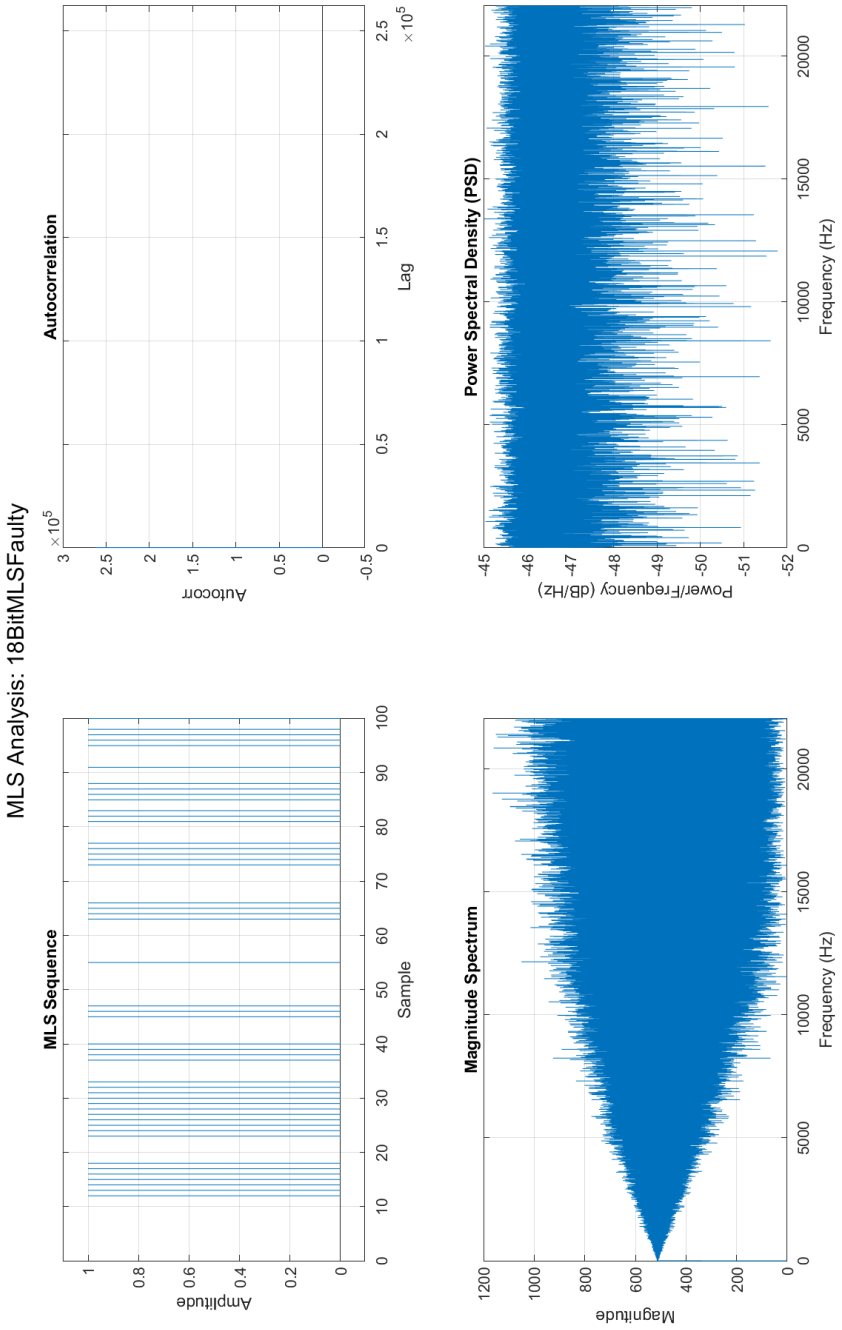


Figure I.42: Plots related to the faulty 18-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 524287 samples
Detected register size: 19 bits
✖Sequence FAILED MLS check.
Length : 524287 samples
Register size (n) : 19 bits
Detected cyclic shift : NaN
Autocorrelation peak : 524287.000000 (ideal = 524287)
Max sidelobe deviation: 4.000000

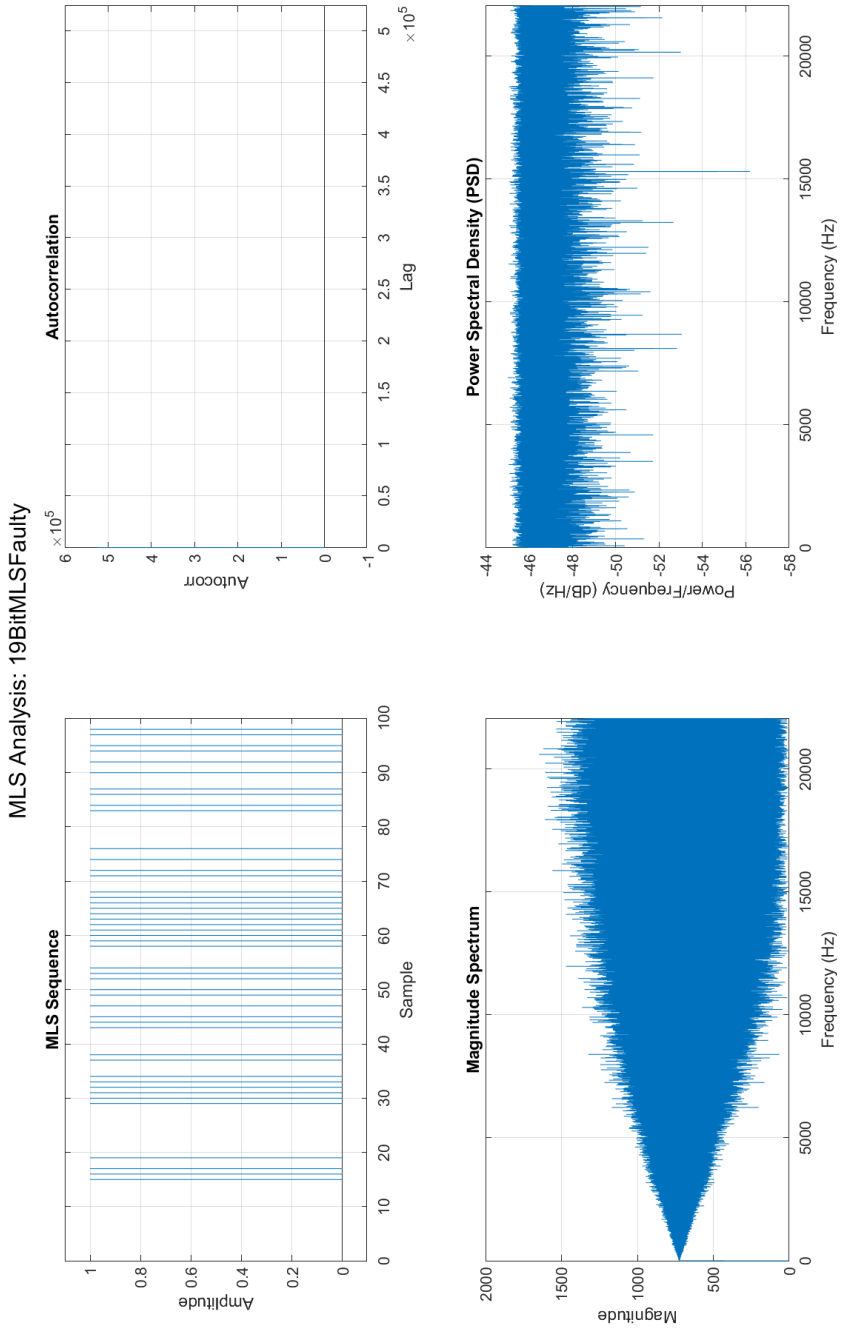


Figure I.43: Plots related to the faulty 19-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 1048575 samples
Detected register size: 20 bits
✖Sequence FAILED MLS check.
Length : 1048575 samples
Register size (n) : 20 bits
Detected cyclic shift : NaN
Autocorrelation peak : 1048575.000000 (ideal = 1048575)
Max sidelobe deviation: 4.000000

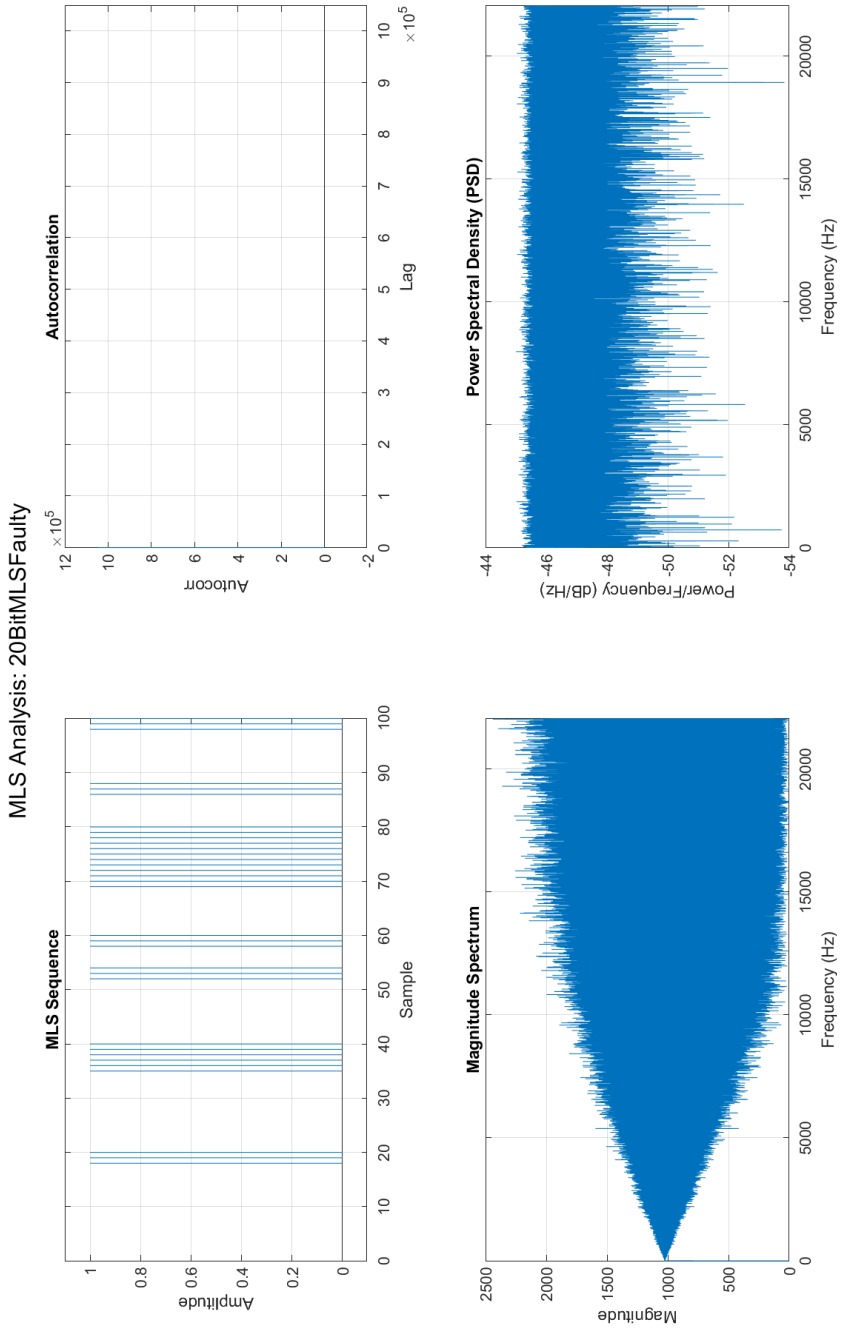


Figure I.44: Plots related to the faulty 20-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 2097151 samples
Detected register size: 21 bits
✖Sequence FAILED MLS check.
Length : 2097151 samples
Register size (n) : 21 bits
Detected cyclic shift : NaN
Autocorrelation peak : 2097151.000000 (ideal = 2097151) Max sidelobe deviation: 4.000000

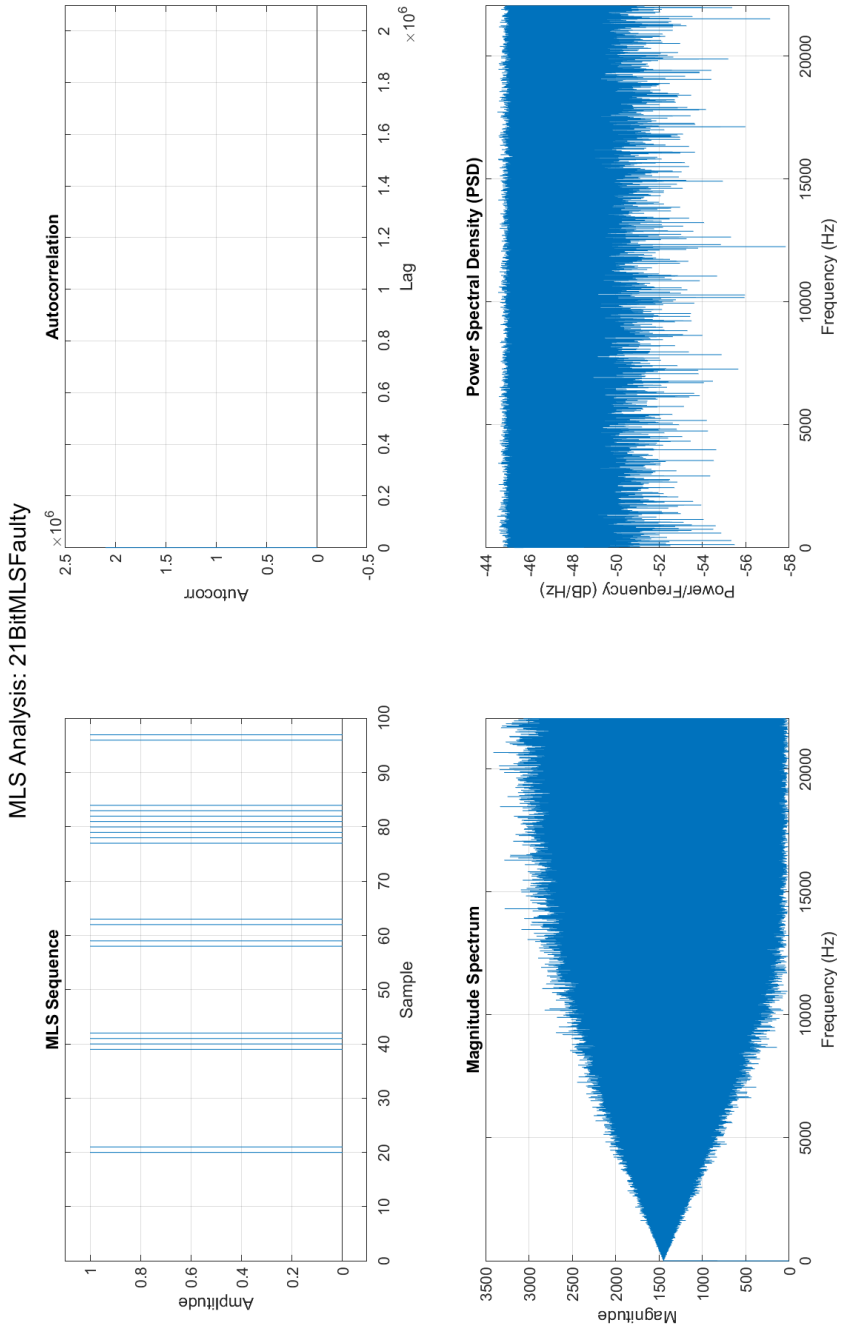


Figure I.45: Plots related to the faulty 21-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 4194303 samples
Detected register size: 22 bits
✖Sequence FAILED MLS check.
Length : 4194303 samples
Register size (n) : 22 bits
Detected cyclic shift : NaN
Autocorrelation peak : 4194303.000000 (ideal = 4194303)
Max sidelobe deviation: 4.000000

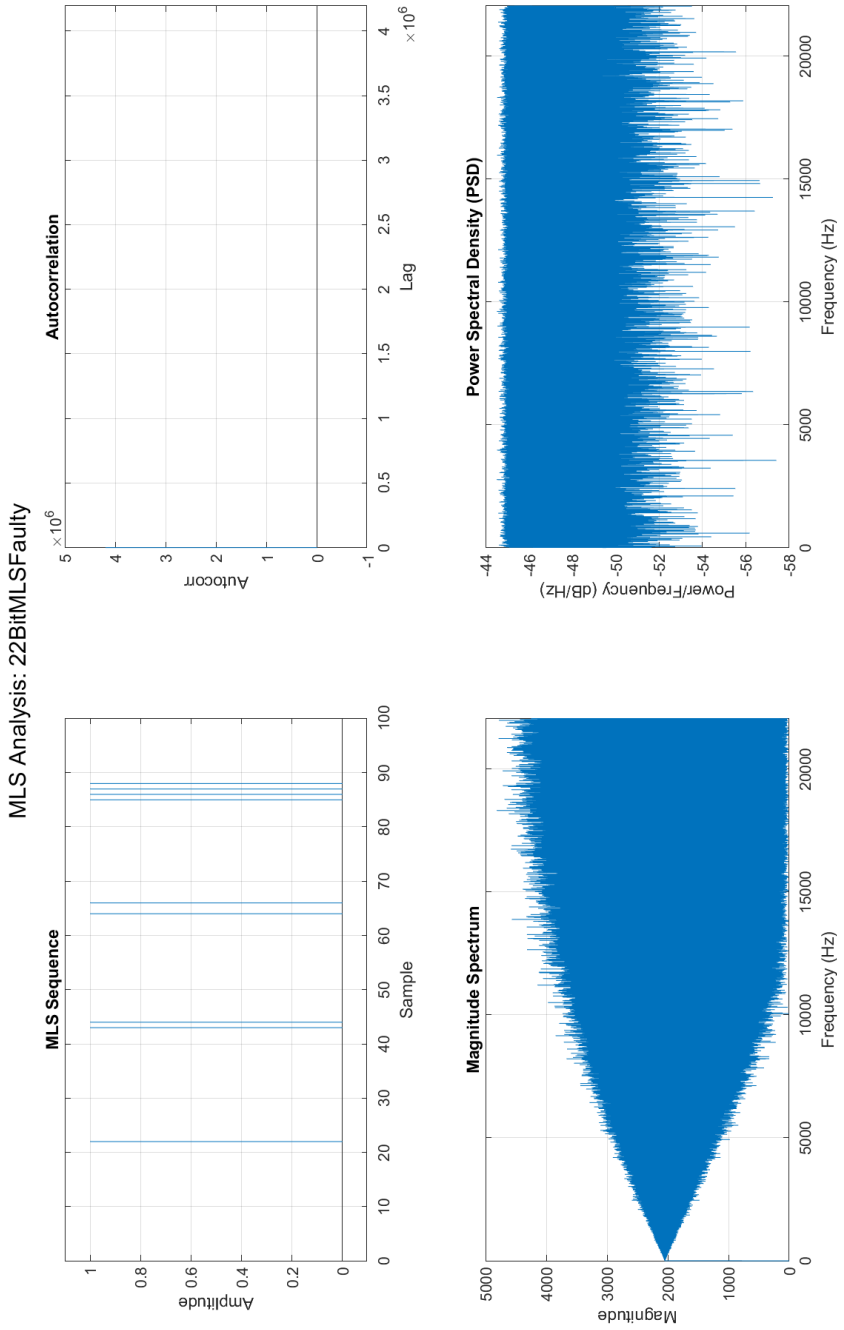


Figure I.46: Plots related to the faulty 22-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 8388607 samples
Detected register size: 23 bits
✖Sequence FAILED MLS check.
Length : 8388607 samples
Register size (n) : 23 bits
Detected cyclic shift : NaN
Autocorrelation peak : 8388607.000000 (ideal = 8388607)
Max sidelobe deviation: 4.000000

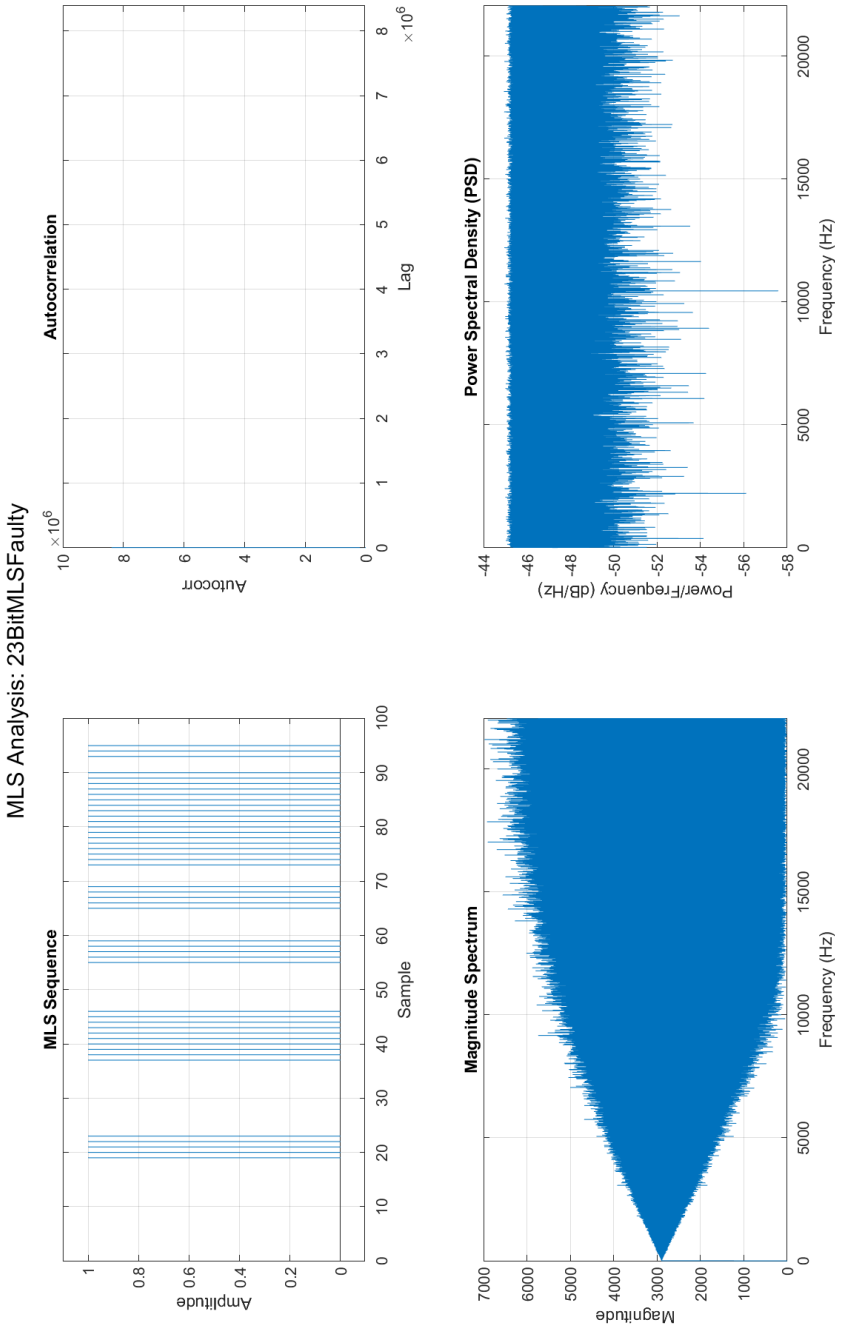


Figure I.47: Plots related to the faulty 23-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 16777215 samples
Detected register size: 24 bits
✖Sequence FAILED MLS check.
Length : 16777215 samples
Register size (n) : 24 bits
Detected cyclic shift : NaN
Autocorrelation peak : 16777215.000000 (ideal = 16777215)
Max sidelobe deviation: 4.000000

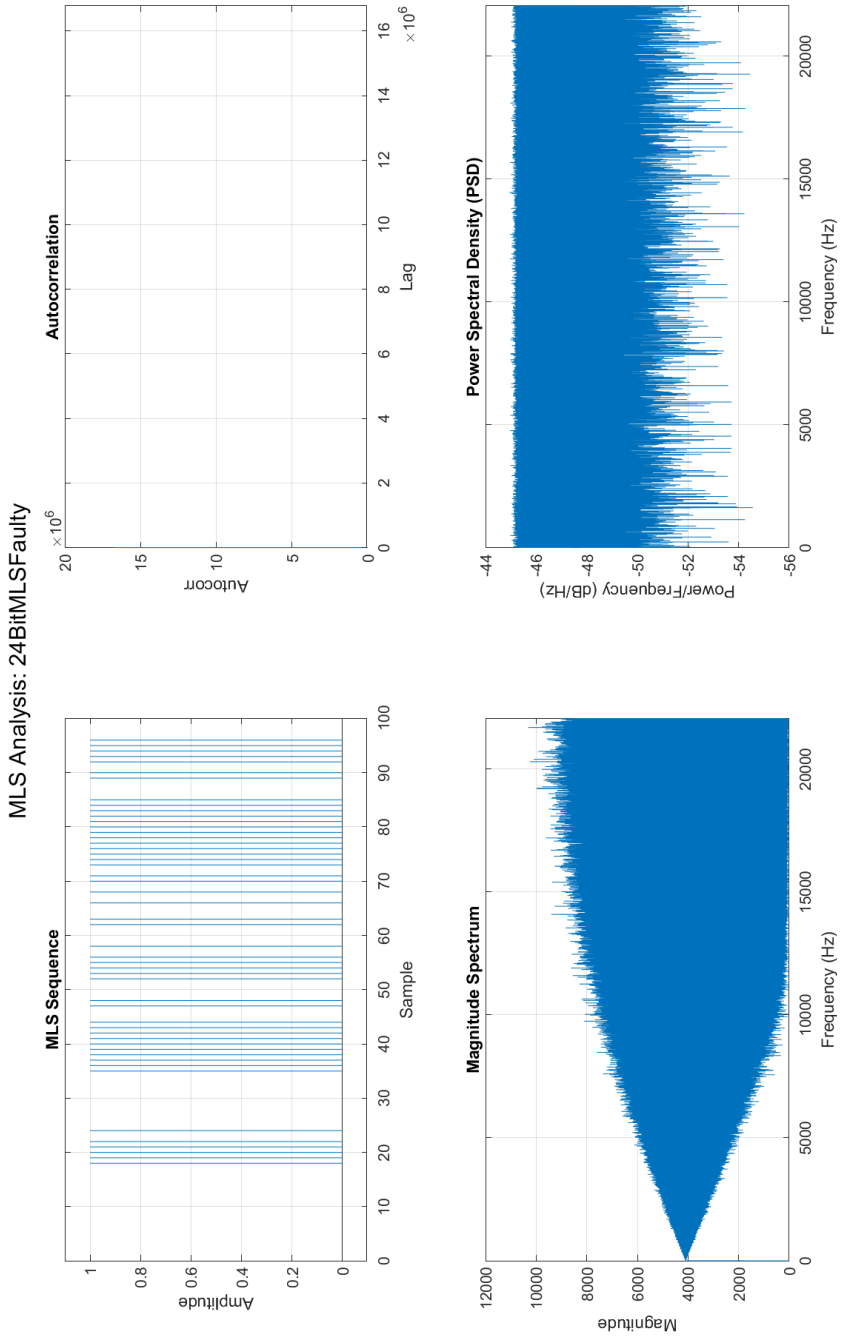


Figure I.48: Plots related to the faulty 24-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 33554431 samples
Detected register size: 25 bits
✖Sequence FAILED MLS check.
Summary:
Length : 33554431 samples
Register size (n) : 25 bits
Detected cyclic shift : NaN
Autocorrelation peak : 33554431.000000 (ideal = 33554431)
Max sidelobe deviation: 4.000000

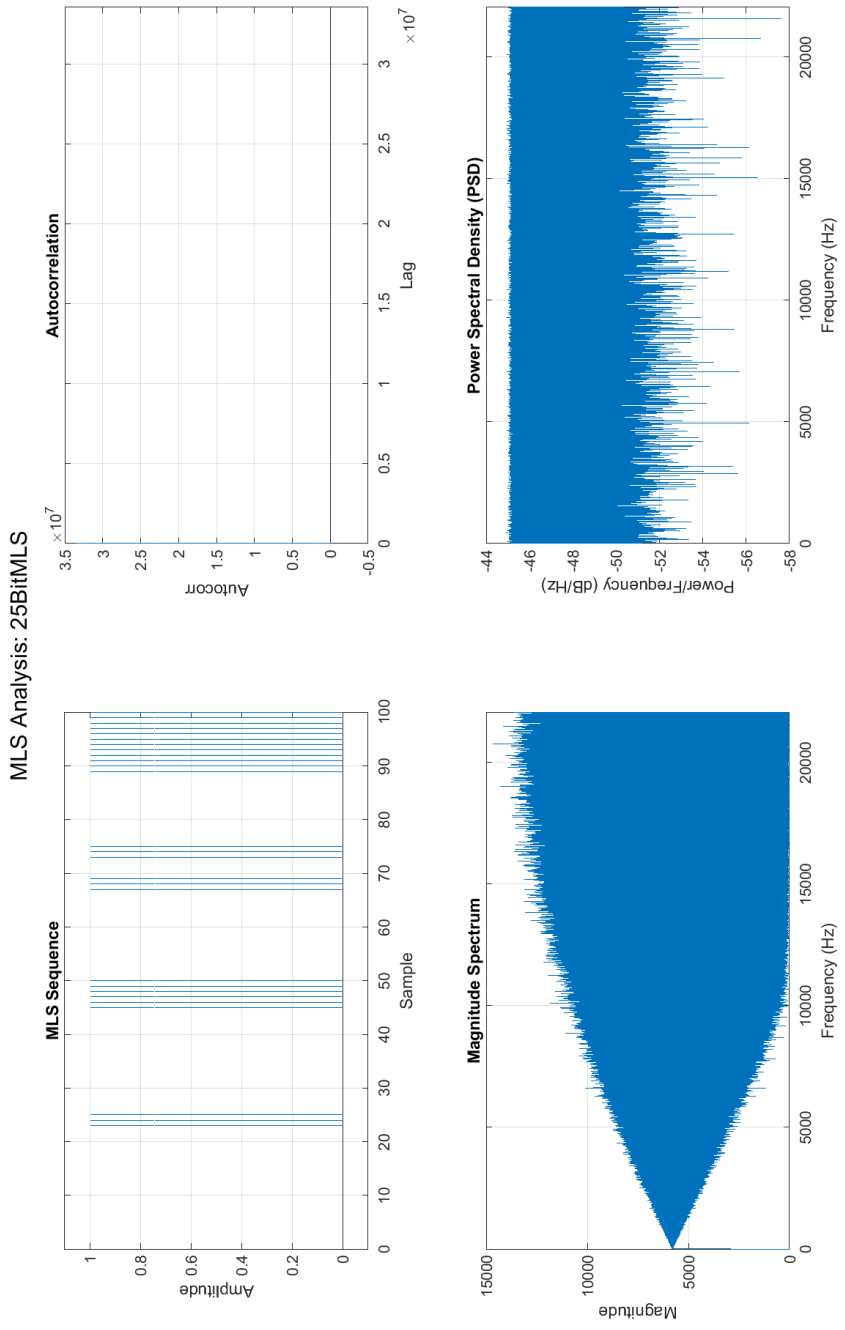


Figure I.49: Plots related to the faulty 25-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 67108863 samples
Detected register size: 26 bits
✖Sequence FAILED MLS check.
Summary:
Length : 67108863 samples
Register size (n) : 26 bits
Detected cyclic shift : NaN
Autocorrelation peak : 67108863.000000 (ideal = 67108863)
Max sidelobe deviation: 4.000000

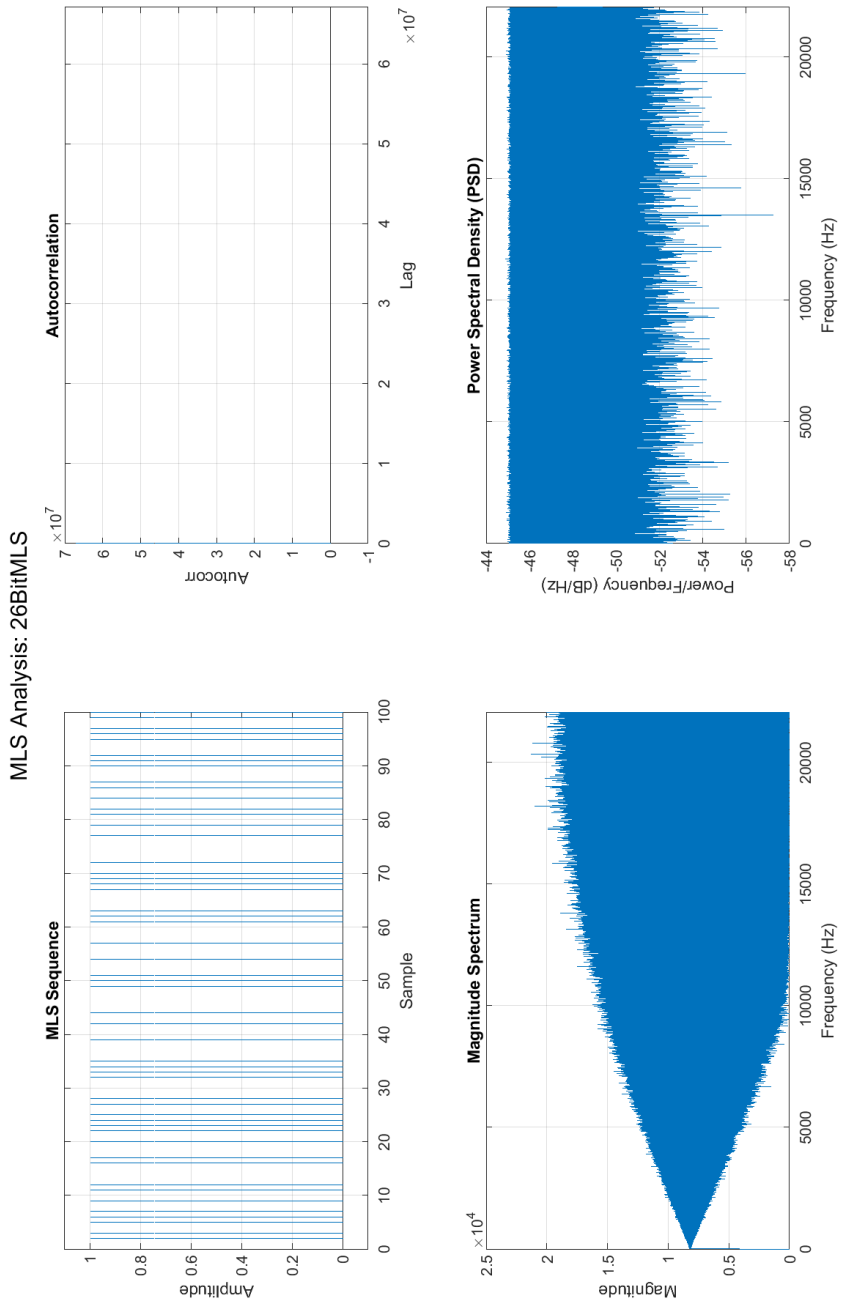


Figure I.50: Plots related to the faulty 26-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 134217727 samples
Detected register size: 27 bits
✖Sequence FAILED MLS check.
Summary:
Length : 134217727 samples
Register size (n) : 27 bits
Detected cyclic shift : NaN
Autocorrelation peak : 134217727.000000 (ideal = 134217727)
Max sidelobe deviation: 4.000000

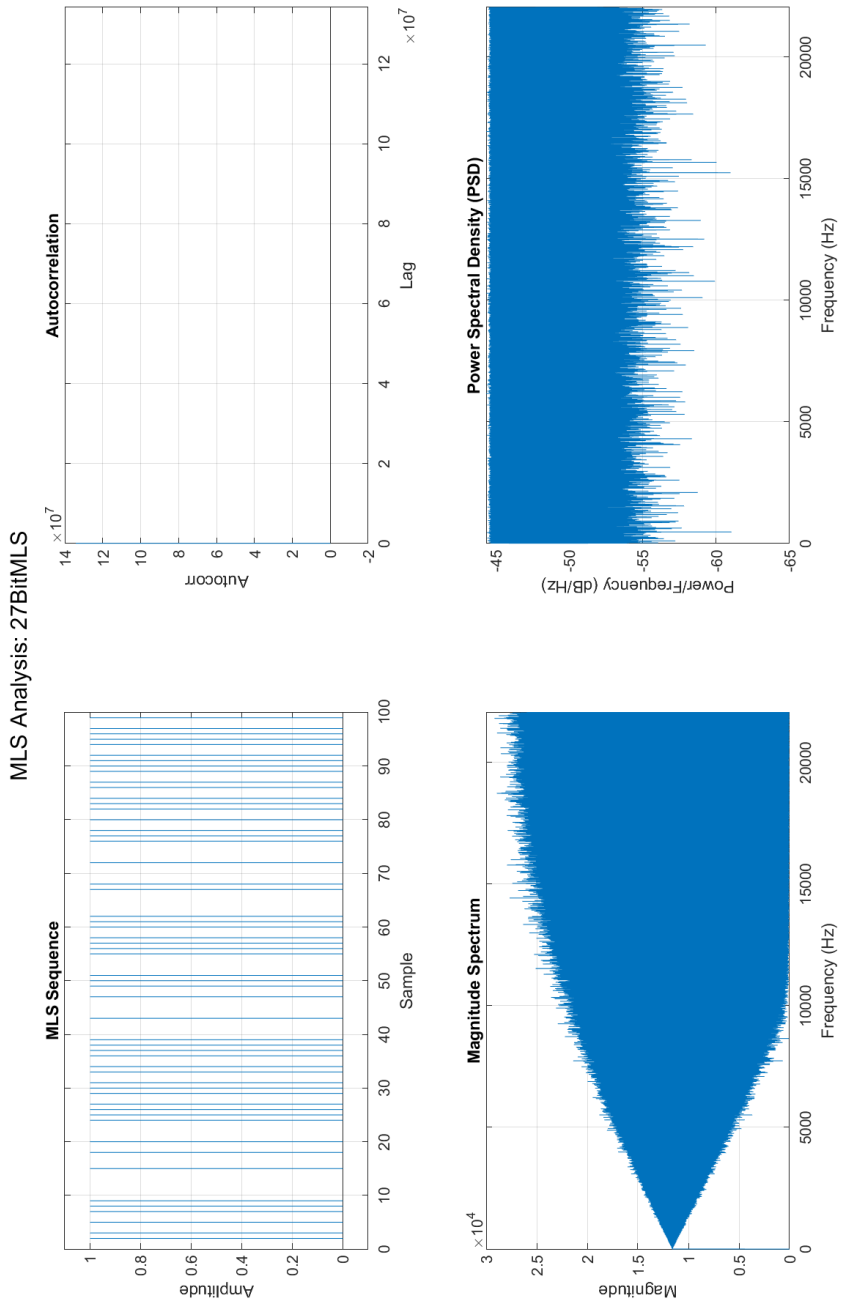


Figure I.51: Plots related to the faulty 27-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 134217727 samples
Detected register size: 28 bits
✖Sequence FAILED MLS check.
Summary:
Length : 268435456 samples
Register size (n) : 28 bits
Detected cyclic shift : NaN
Autocorrelation peak : 268435456.000000 (ideal = 268435456)
Max sidelobe deviation: 4.000000

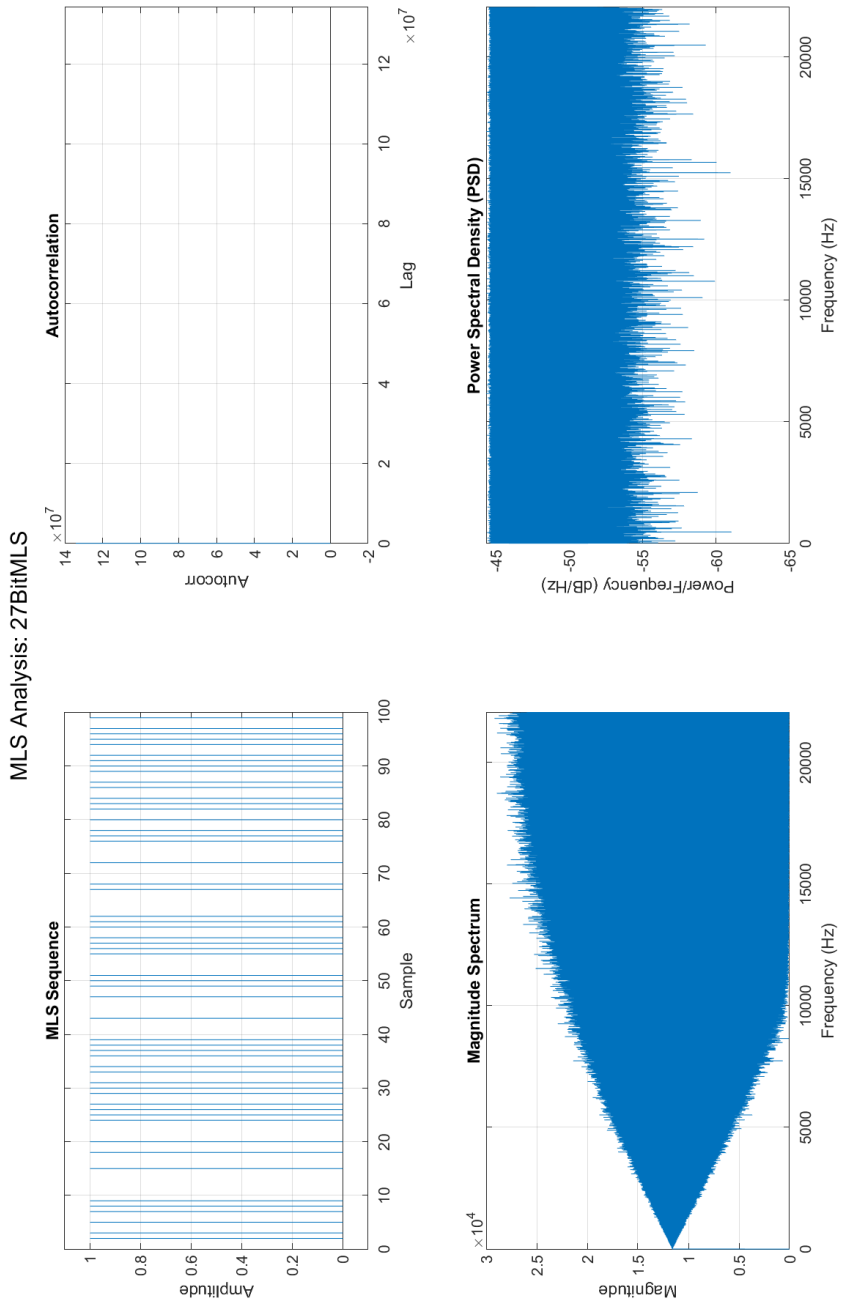


Figure I.52: Plots related to the faulty 28-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

Sequence length: 536870911 samples
Detected register size: 29 bits
✖Sequence FAILED MLS check.
Summary:
Length : 536870911 samples
Register size (n) : 29 bits
Detected cyclic shift : NaN
Autocorrelation peak : 536870911.000000 (ideal = 536870911)
Max sidelobe deviation: 0.000000

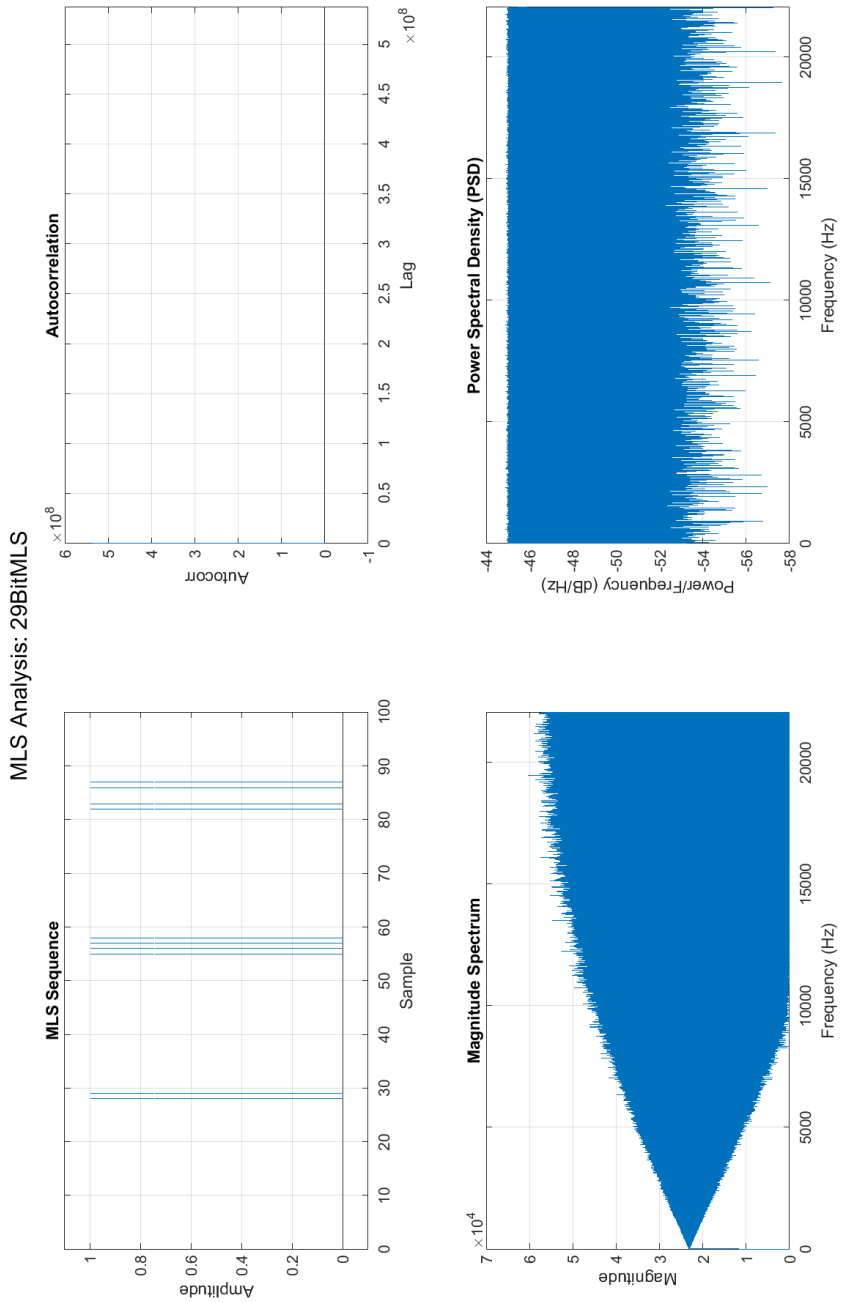


Figure I.53: Plots related to the faulty 29-bit MLS sequence generated on the Teensy. Notice that even though it is faulty, it isn't obvious from a plot.

J Frequency Domain Plots Individual Microphones

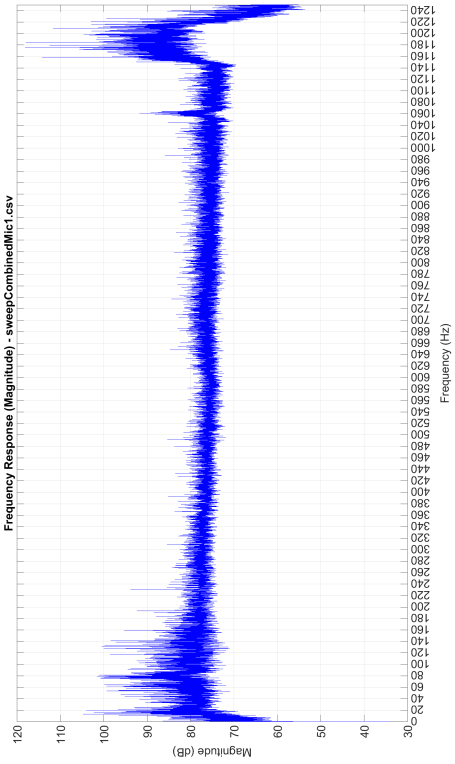
As mentioned in section 5.3.2, an issue arose when testing two microphones at once, leading to a suspicion towards if one or more microphones might be defective. Therefore, a complete set of impulse responses has been measured for each microphone. In figures J.1-J.8, said impulse responses in the frequency domain can be seen for all the available microphones. All 8 microphones have been tested using the same program, with the only difference being the suffix before saving data as .csv files. The program used to derive impulse responses is the same as the one used in section 5.3.1, and can be found on GitHub here or in the attached files as "XXXX".

By examining figure J.1-J.8, it is clear that the microphones does not have identical impulse responses, unless the fault can be found in the speaker used, and discrepancies in how the microphone was placed during testing, despite best efforts being made to place all 8 identically. The general test setup and test procedure are identical to the one used in section 5.3.1 on page 5.3.1.

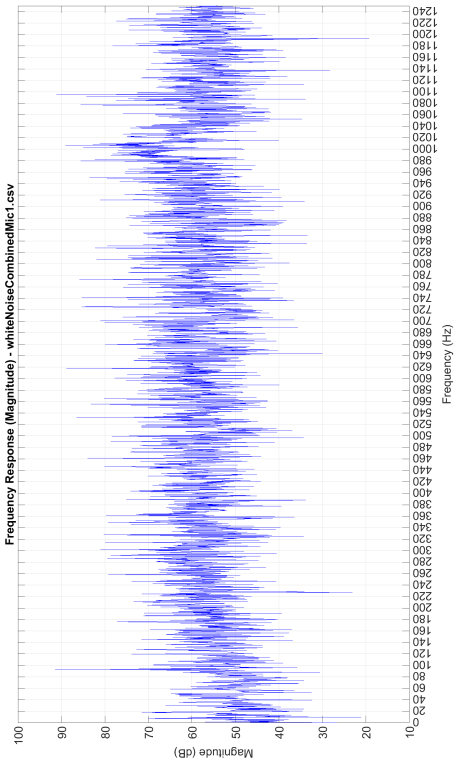
Based on figure J.1-J.8, the microphones are subjectively ranked as follows:

- Microphone 8
- Microphone 1
- Microphone 3
- Microphone 5
- Microphone 6
- Microphone 4
- Microphone 7
- Microphone 2

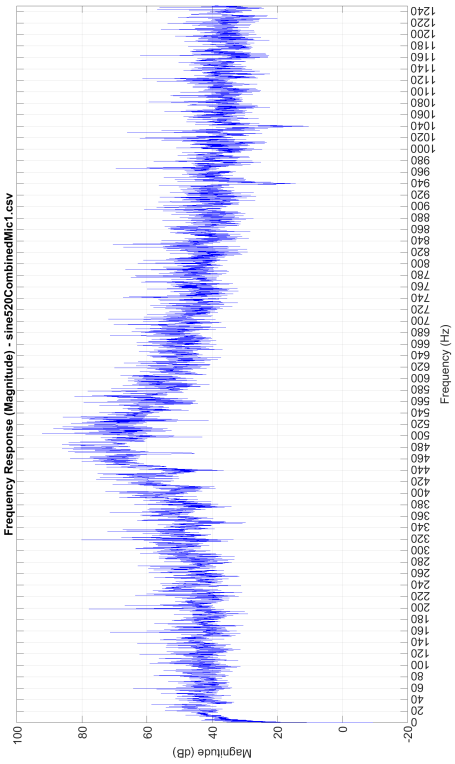
The ranking is purely subjective based on which plots look the most as expected. As time doesn't allow sourcing new microphones, nor testing/calibrating them to ensure identical operation, the project will continue with these impulse responses in mind.



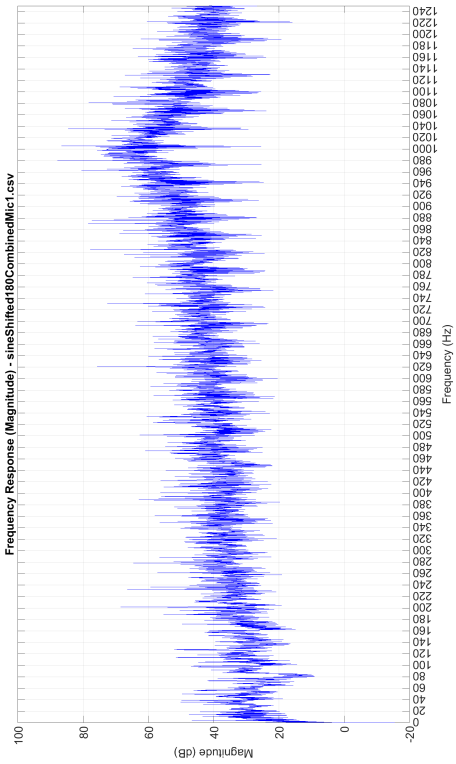
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

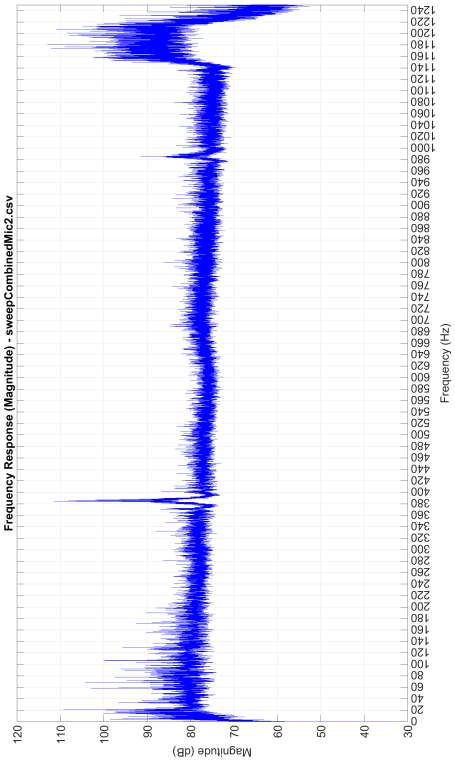


(c) 520Hz puretone sinusoid frequency domain impulse response.

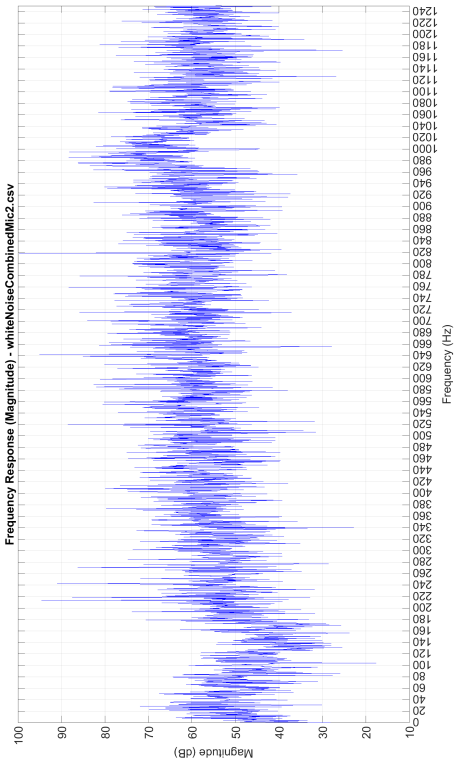


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

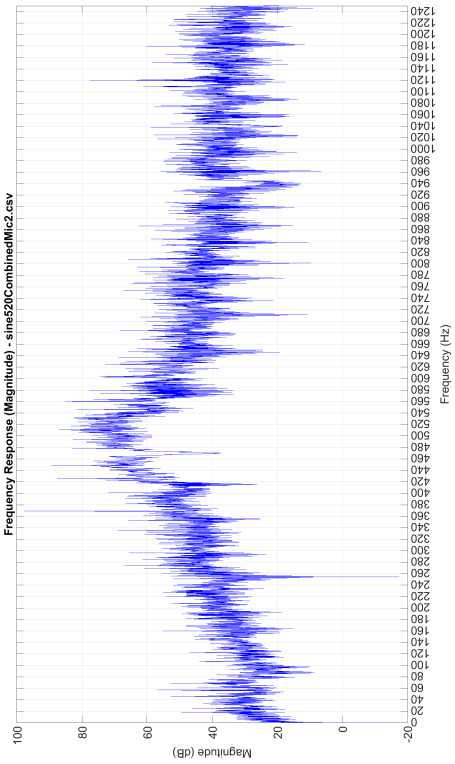
Figure J.1: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 1. All plots span 0-1220Hz.



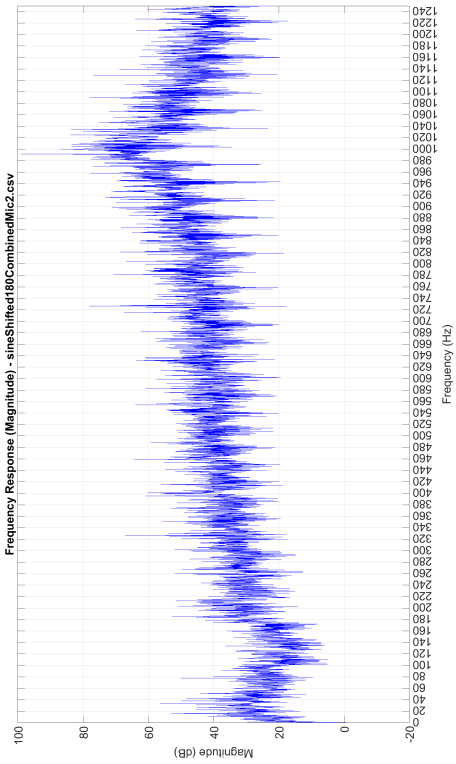
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

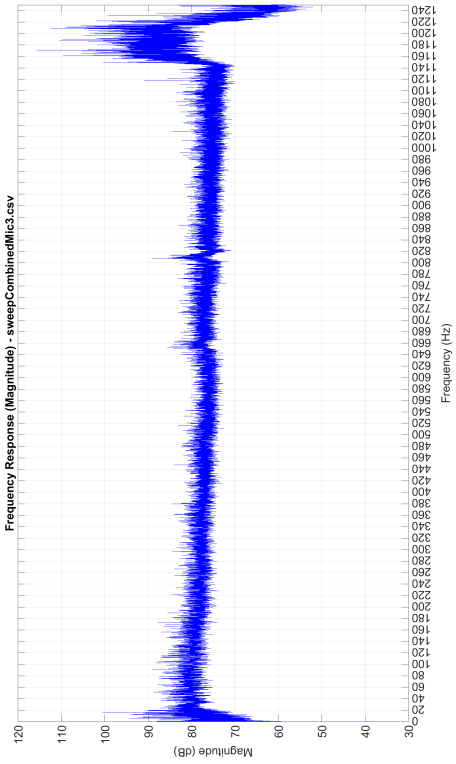


(c) 520Hz puretone sinusoid frequency domain impulse response.

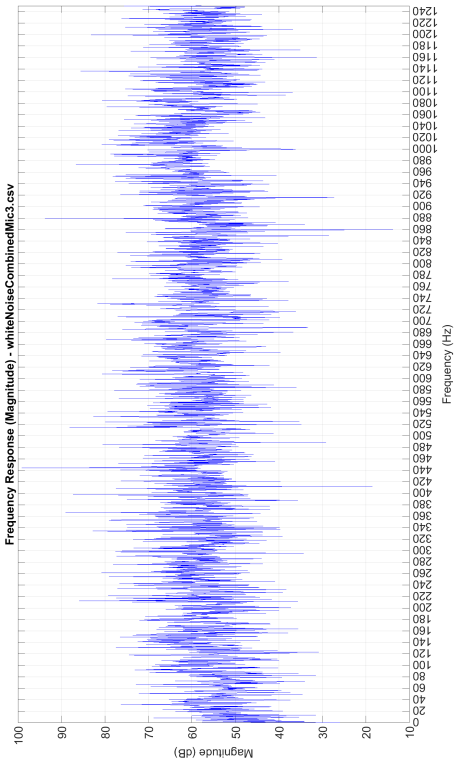


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

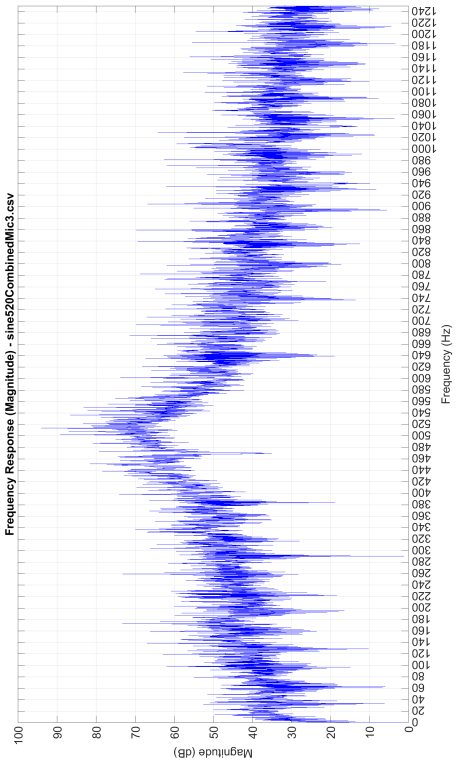
Figure J.2: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 2. All plots span 0-1220Hz.



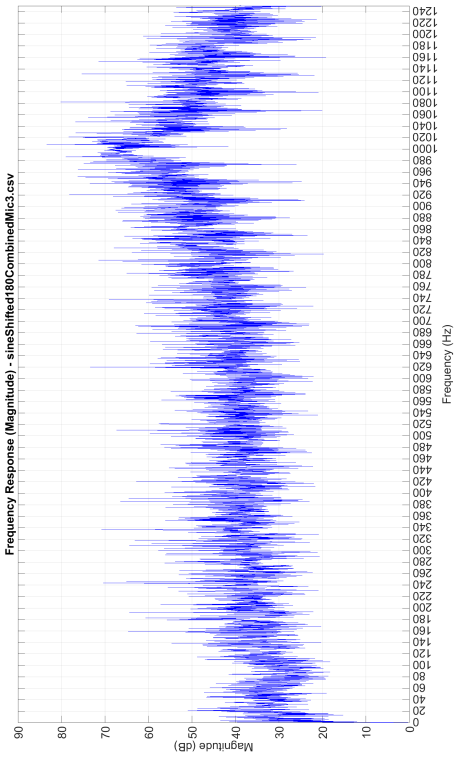
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

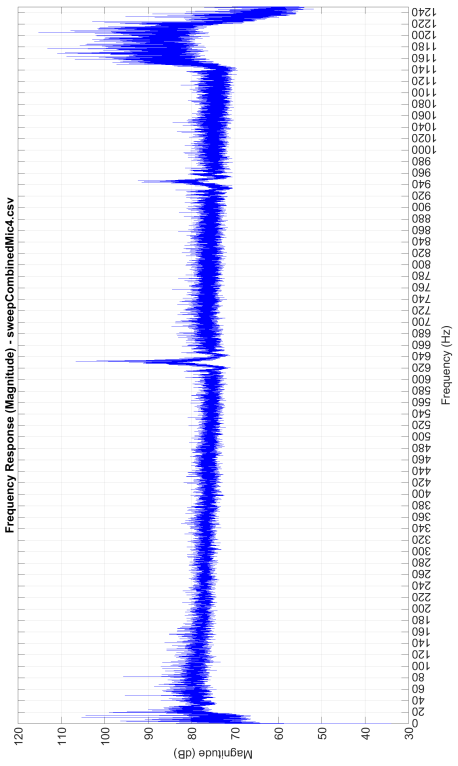


(c) 520Hz puretone sinusoid frequency domain impulse response.

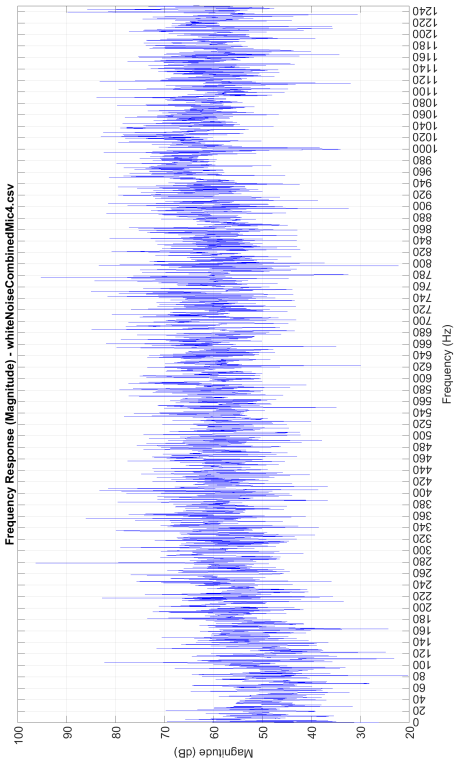


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

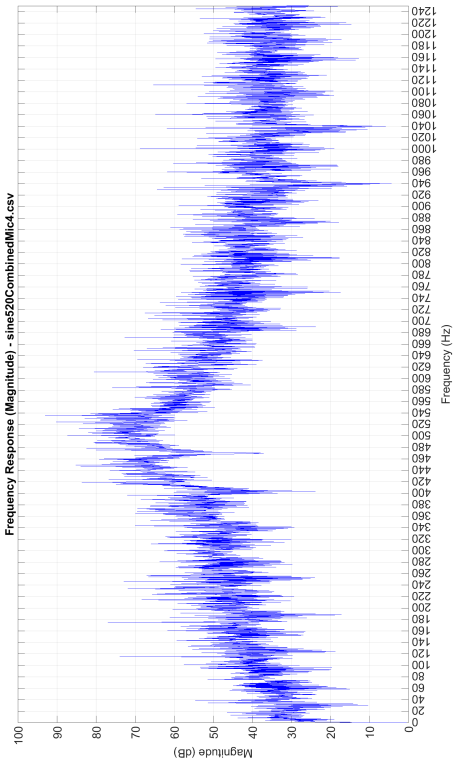
Figure J.3: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 3. All plots span 0-1220Hz.



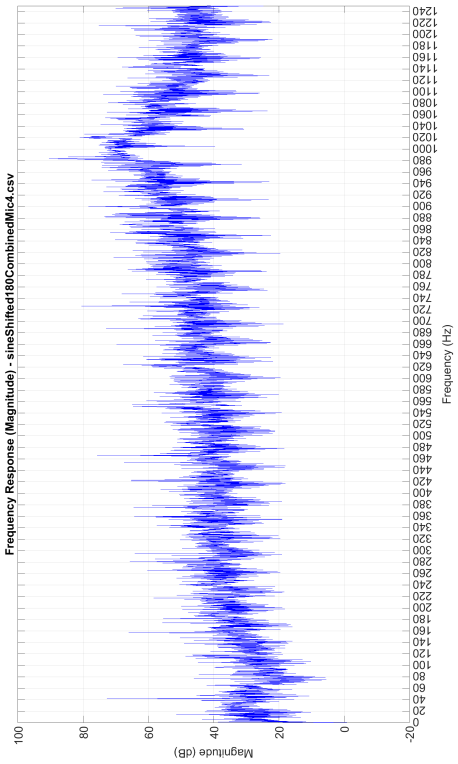
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

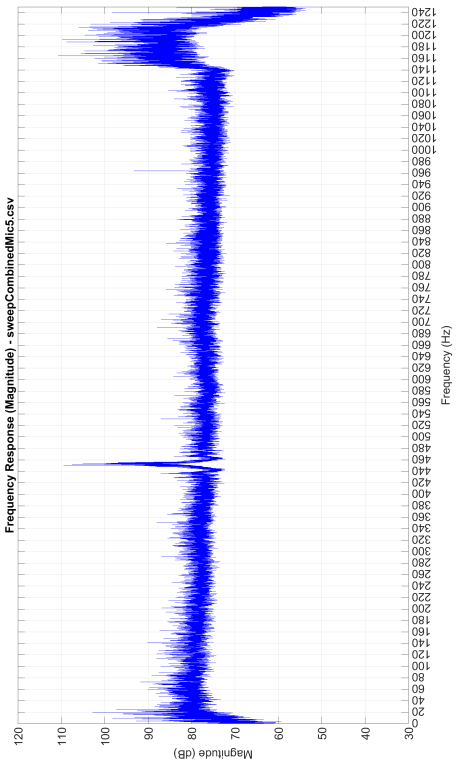


(c) 520Hz puretone sinusoid frequency domain impulse response.

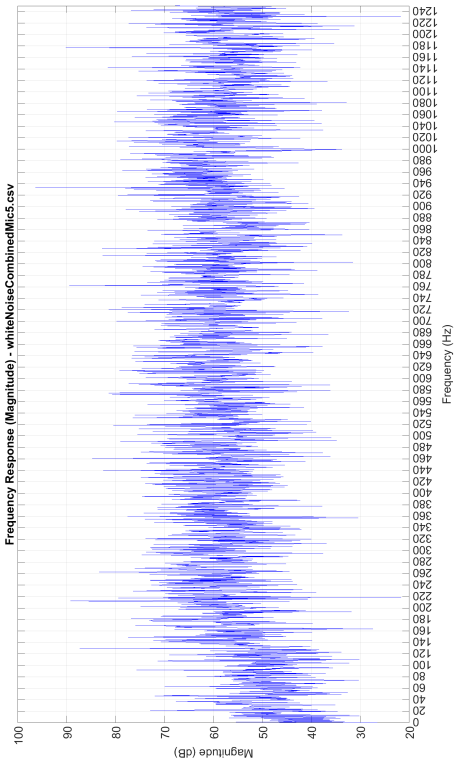


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

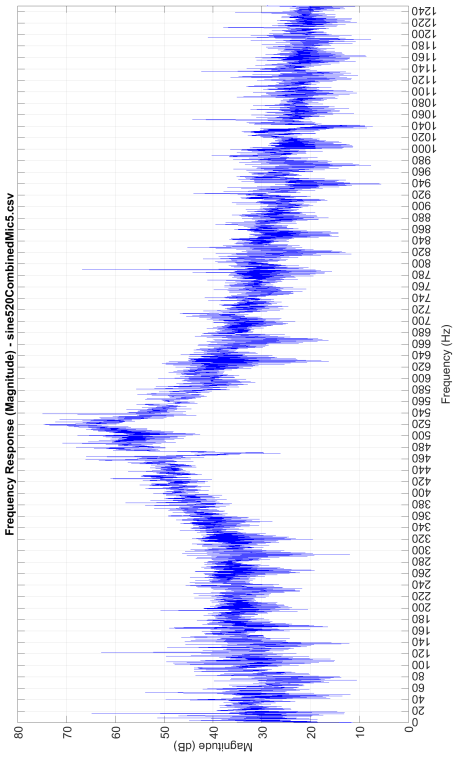
Figure J.4: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 4. All plots span 0-1220Hz.



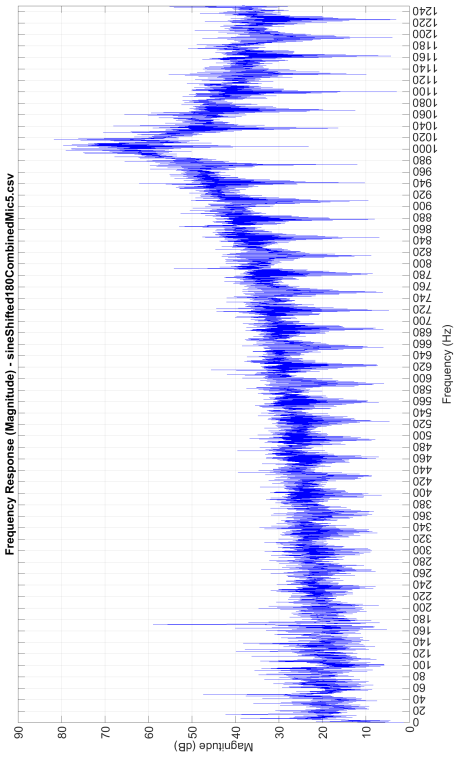
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

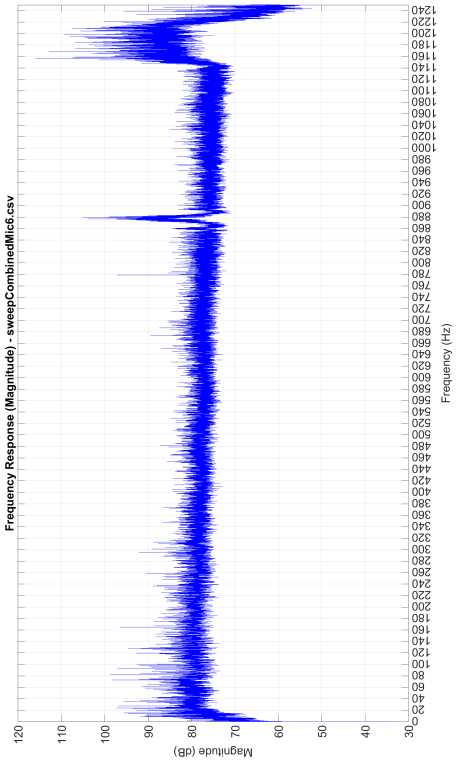


(c) 520Hz puretone sinusoid frequency domain impulse response.

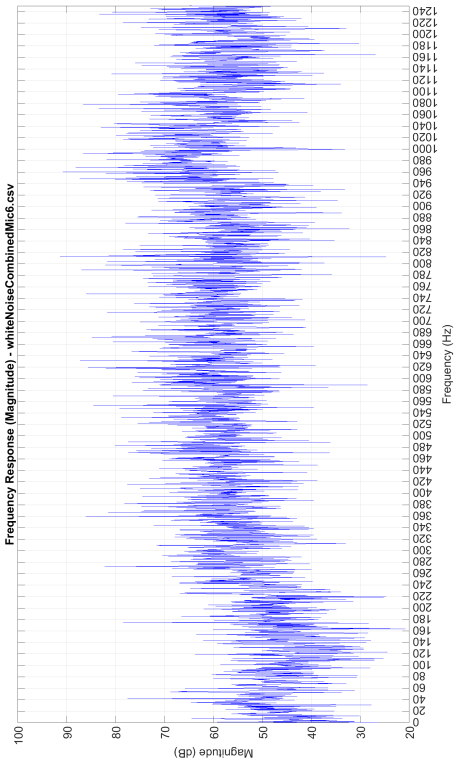


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

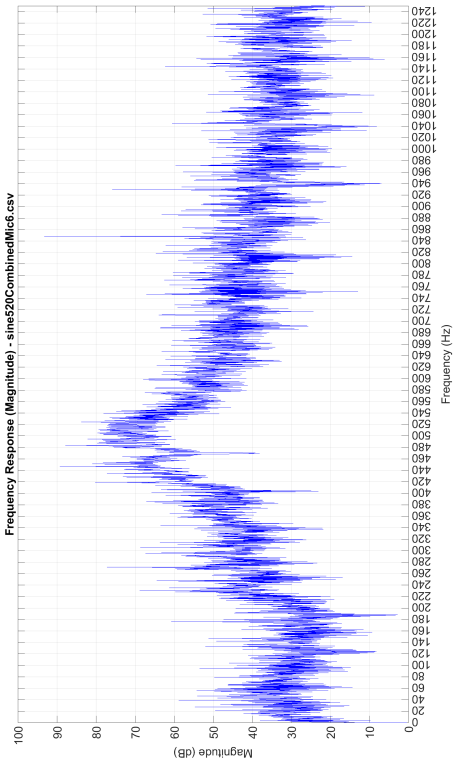
Figure J.5: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 5. All plots span 0-1220Hz.



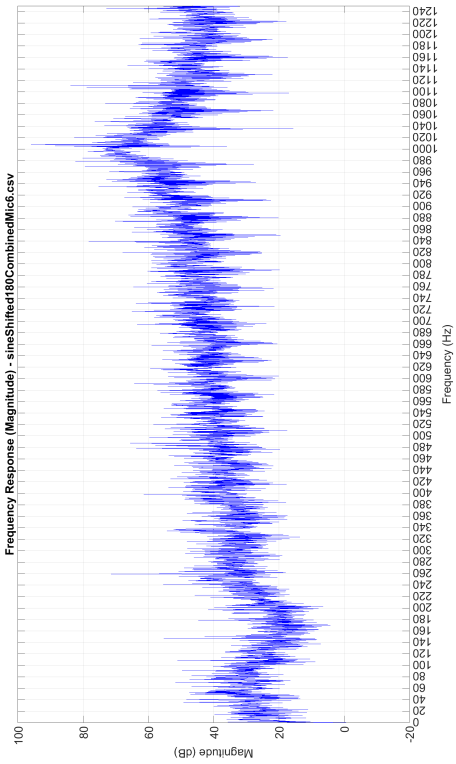
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

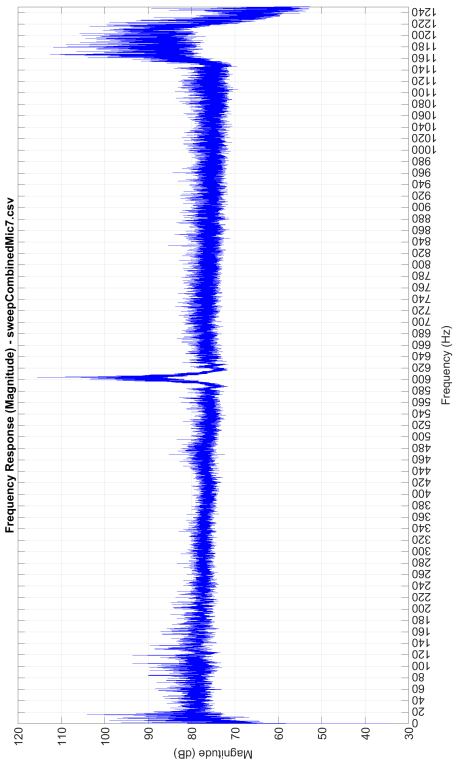


(c) 520Hz puretone sinusoid frequency domain impulse response.

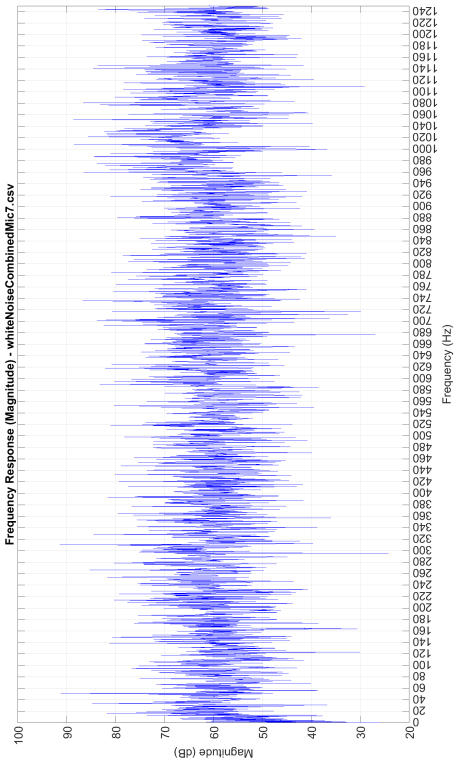


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

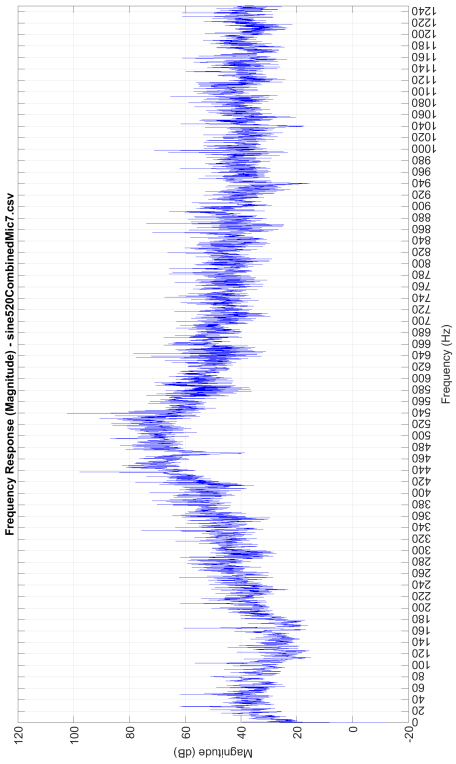
Figure J.6: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 6. All plots span 0-1220Hz.



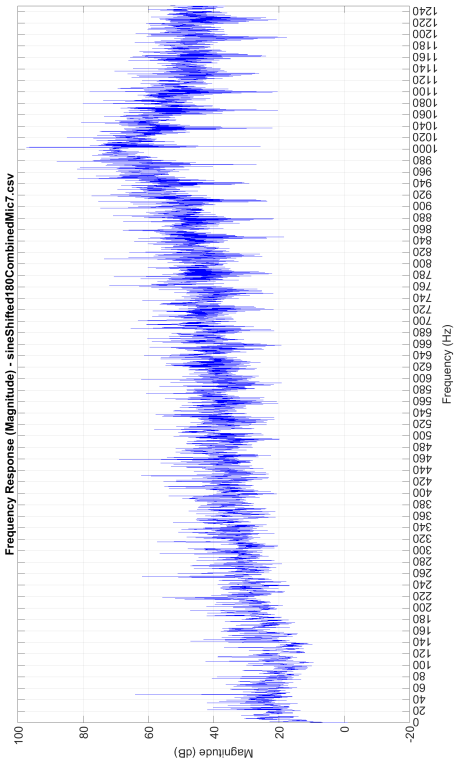
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.

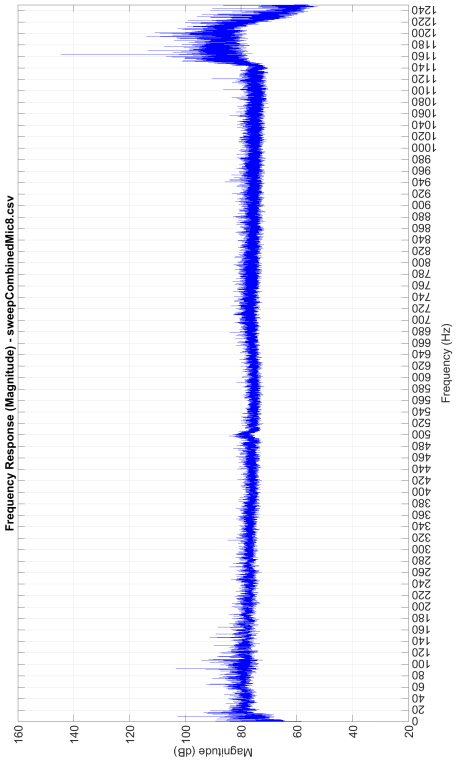


(c) 520Hz puretone sinusoid frequency domain impulse response.

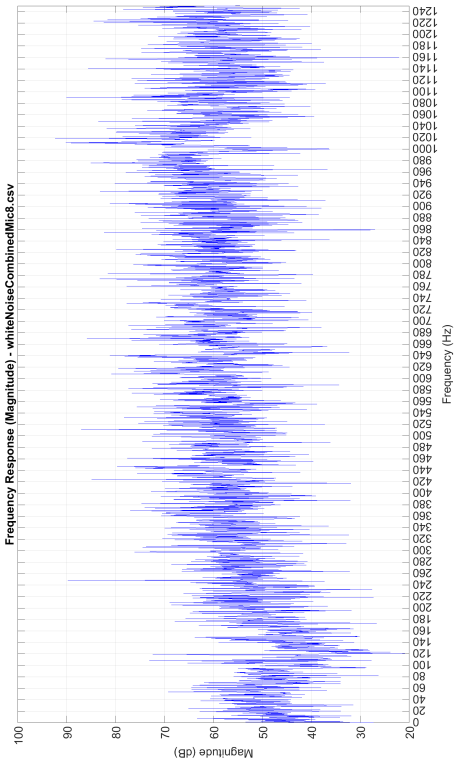


(d) 1kHz phase shifted sinusoid frequency domain impulse response.

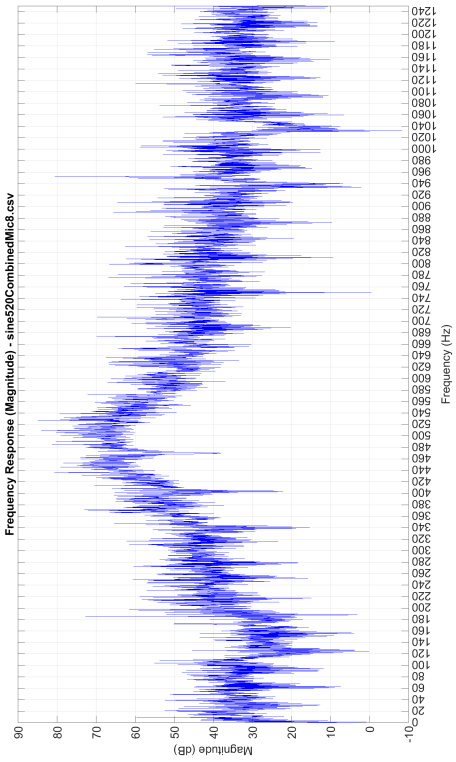
Figure J.7: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 7. All plots span 0-1220Hz.



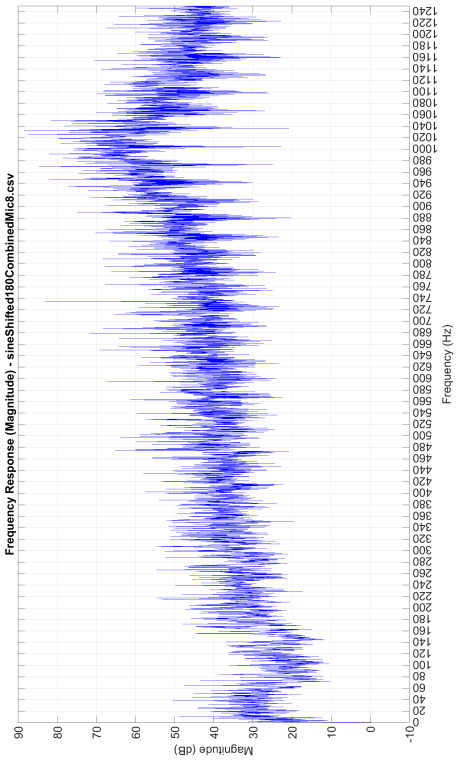
(a) Sine sweep from 20Hz to 1220Hz frequency domain impulse response.



(b) White noise frequency domain impulse response.



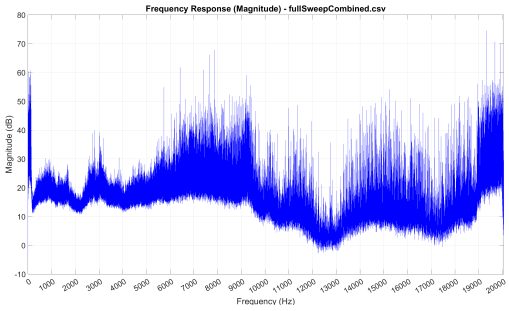
(c) 520Hz puretone sinusoid frequency domain impulse response.



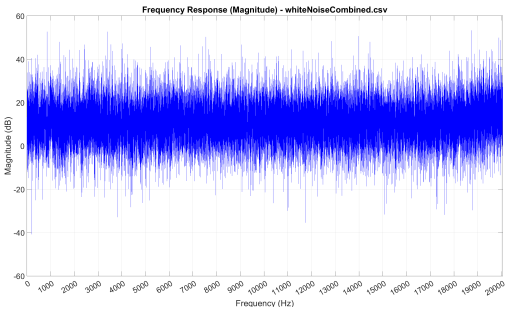
(d) 1kHz phase shifted sinusoid frequency domain impulse response.

Figure J.8: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy for microphone 8. All plots span 0-1220Hz.

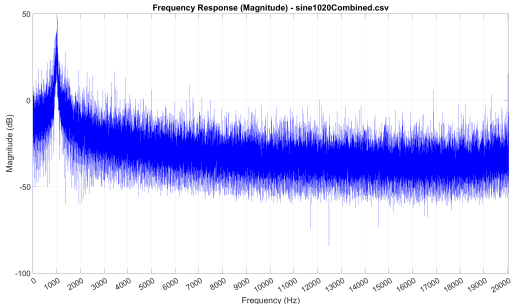
K | Frequency Domain Plots 1 Microphone



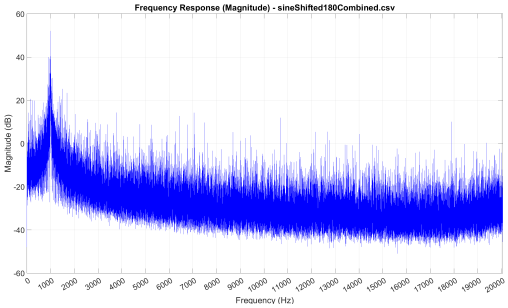
(a) Sine sweep from 20Hz to 20kHz frequency domain impulse response.



(b) White noise frequency domain impulse response.



(c) 1020Hz puretone sinusoid frequency domain impulse response.



(d) 1kHz phase shifted sinusoid frequency domain impulse response.

Figure K.1: Plots of frequency domain impulse responses derived from the impulse responses measured by the Teensy. All plots span 0-20kHz.

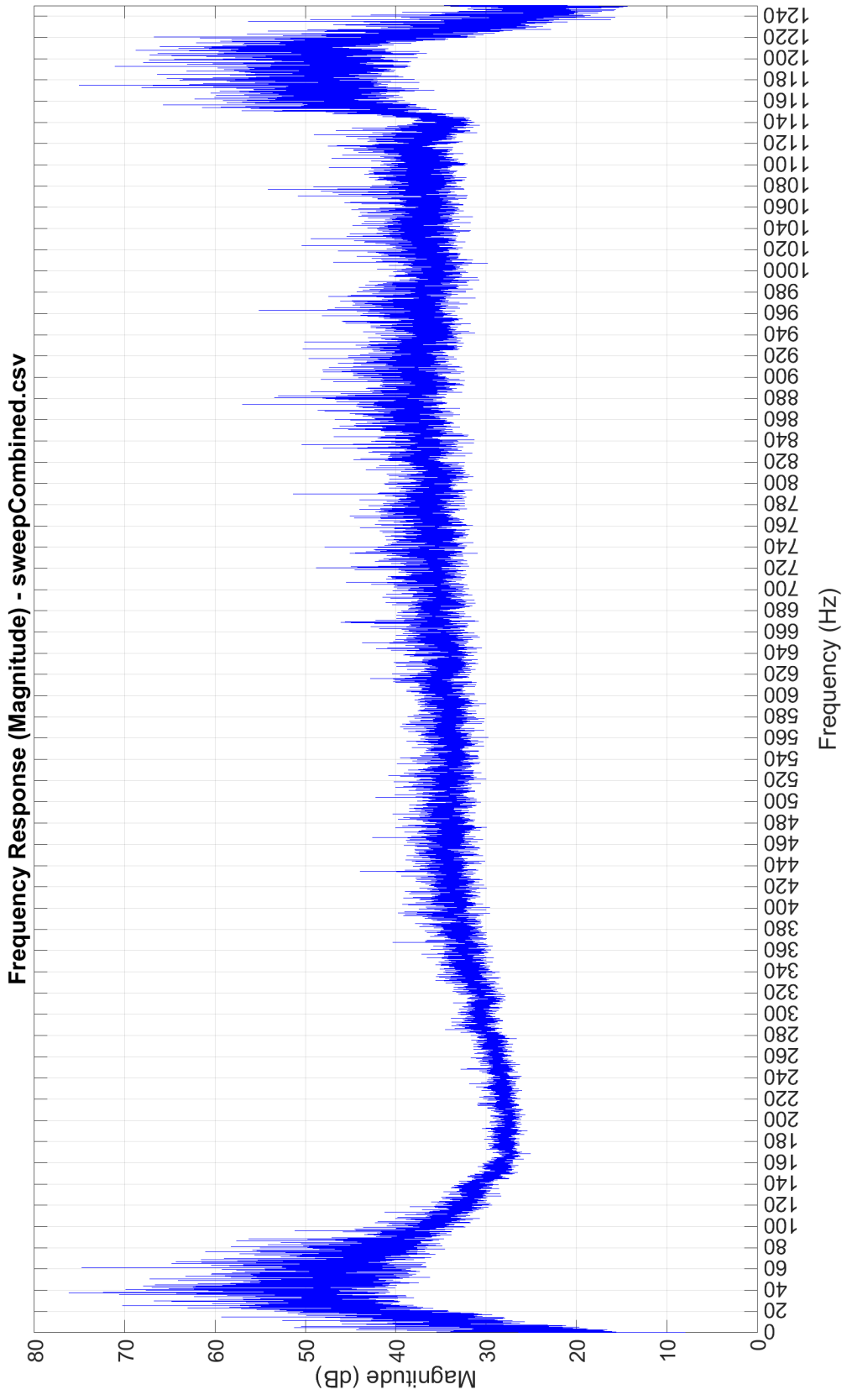


Figure K.2: Sine sweep from 20Hz to 1220Hz frequency domain impulse response.

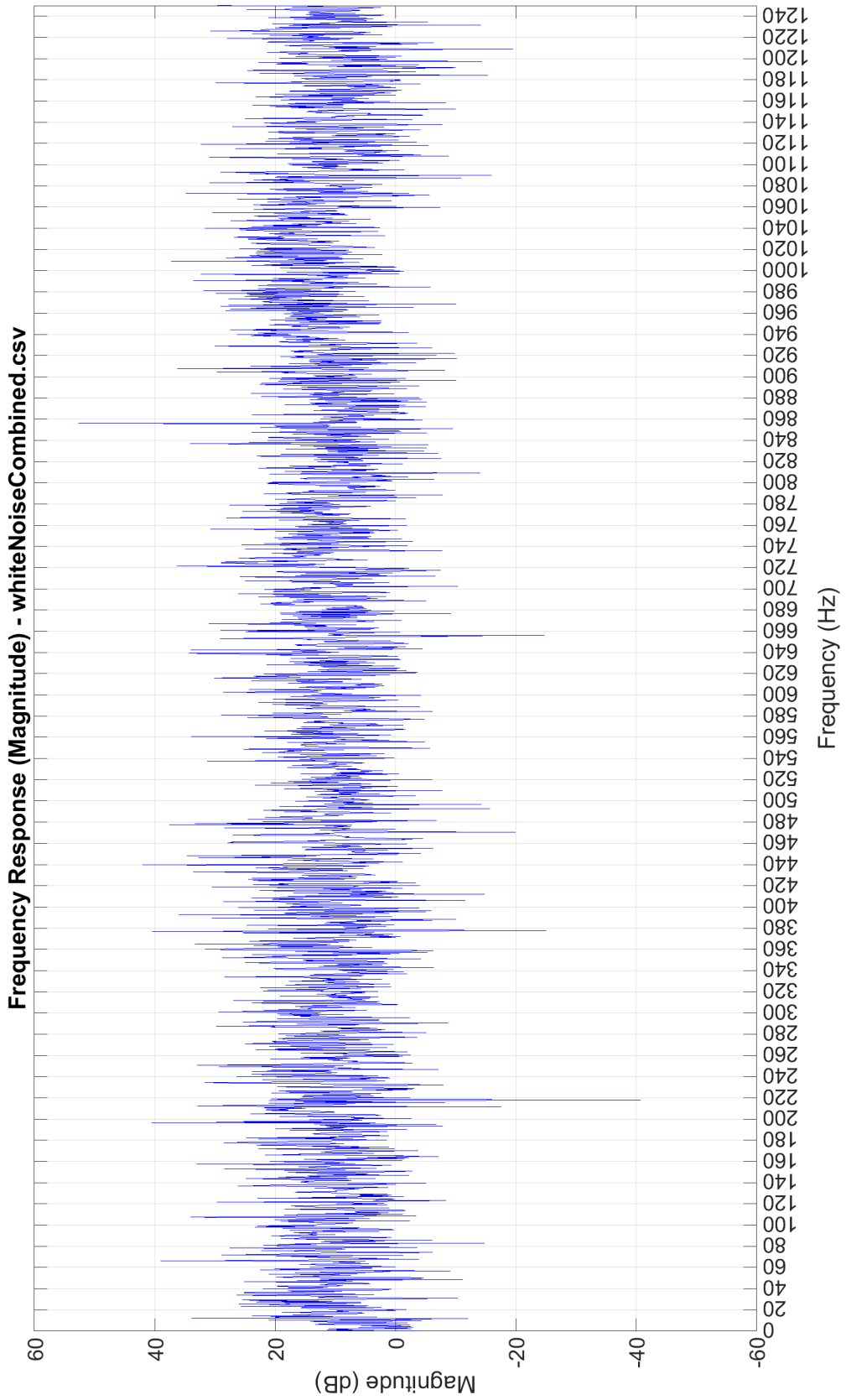


Figure K.3: White noise frequency domain impulse response.

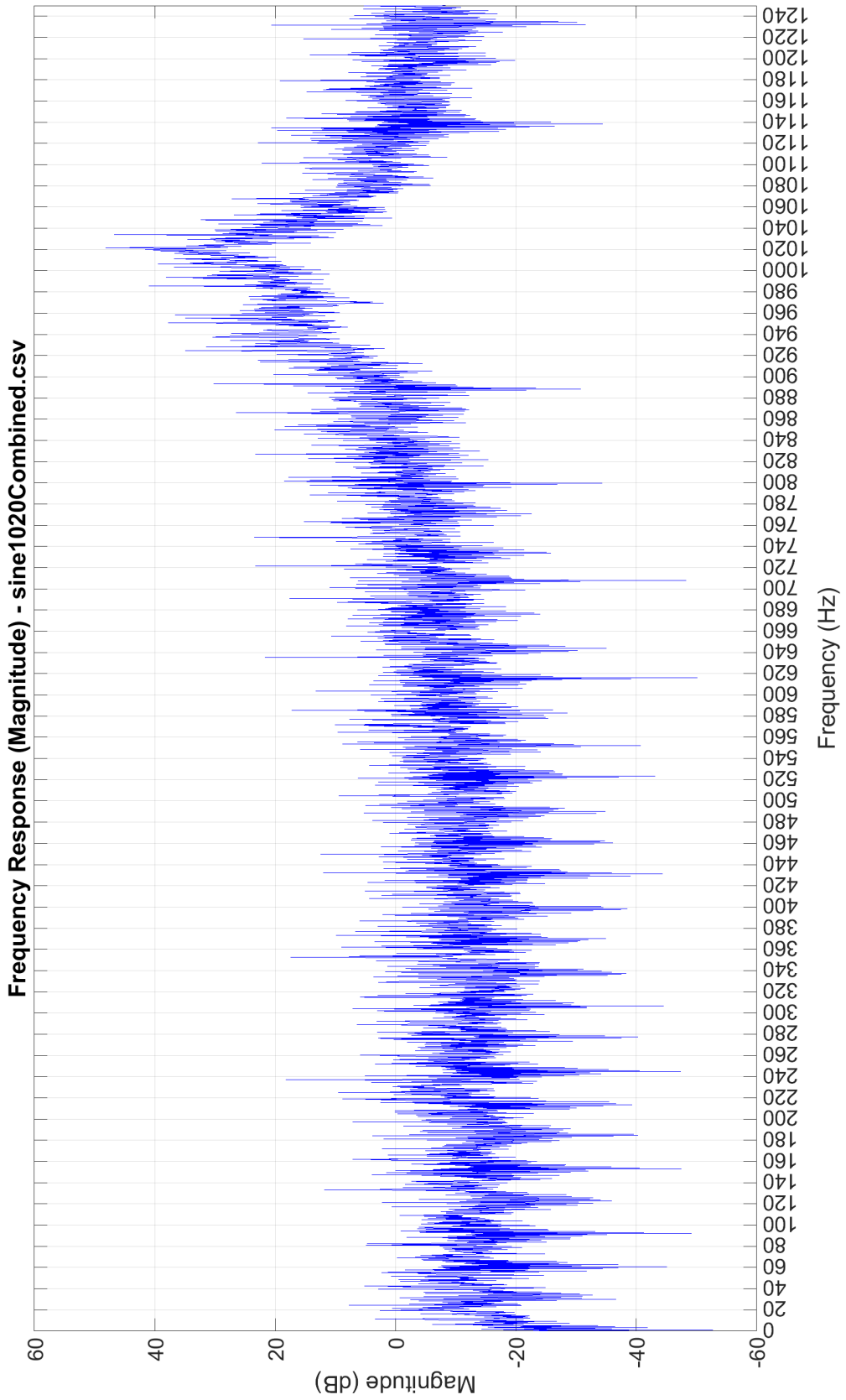


Figure K.4: 1020Hz puretone sinusoid frequency domain impulse response.

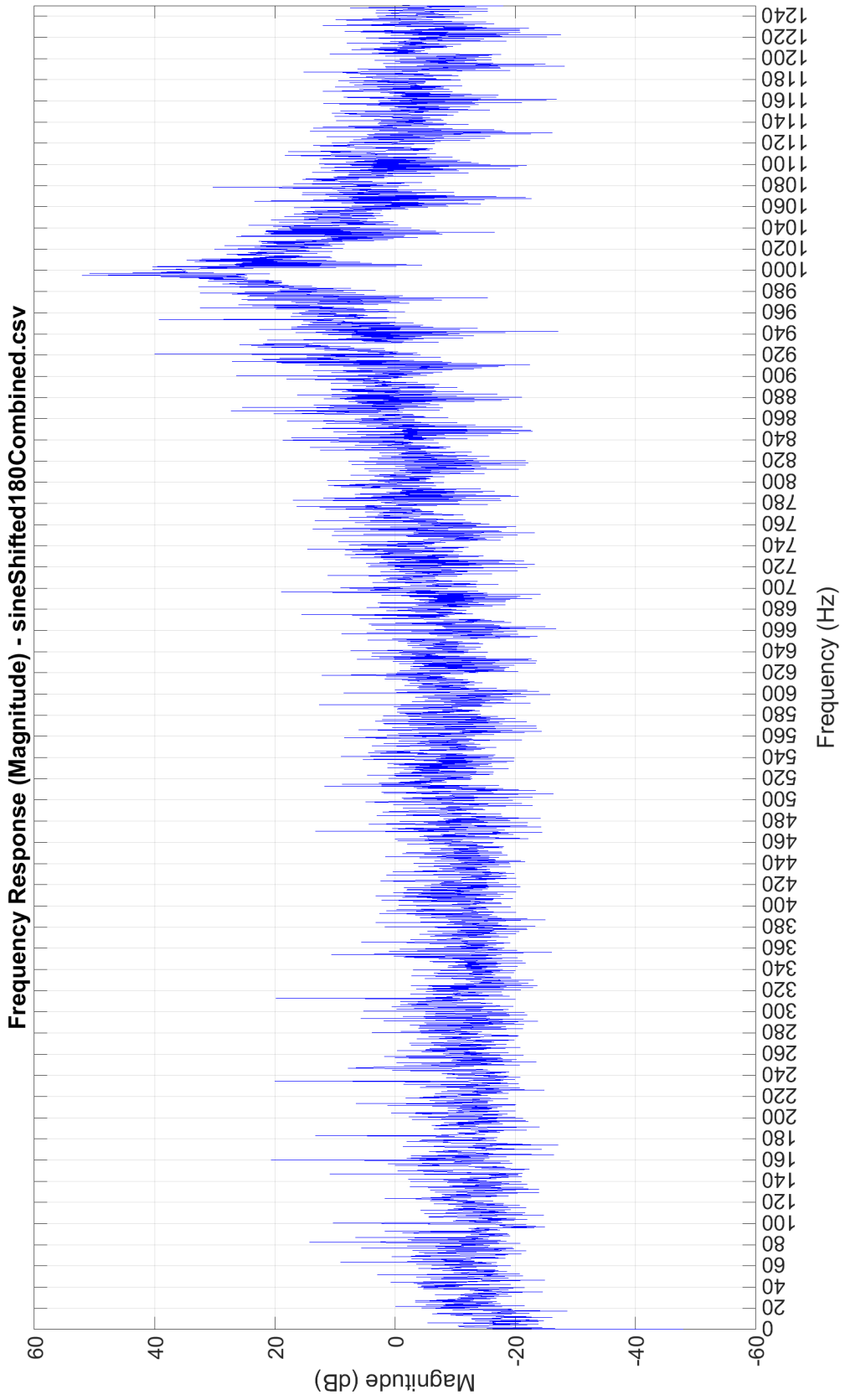


Figure K.5: 1kHz phase shifted sinusoid frequency domain impulse response.

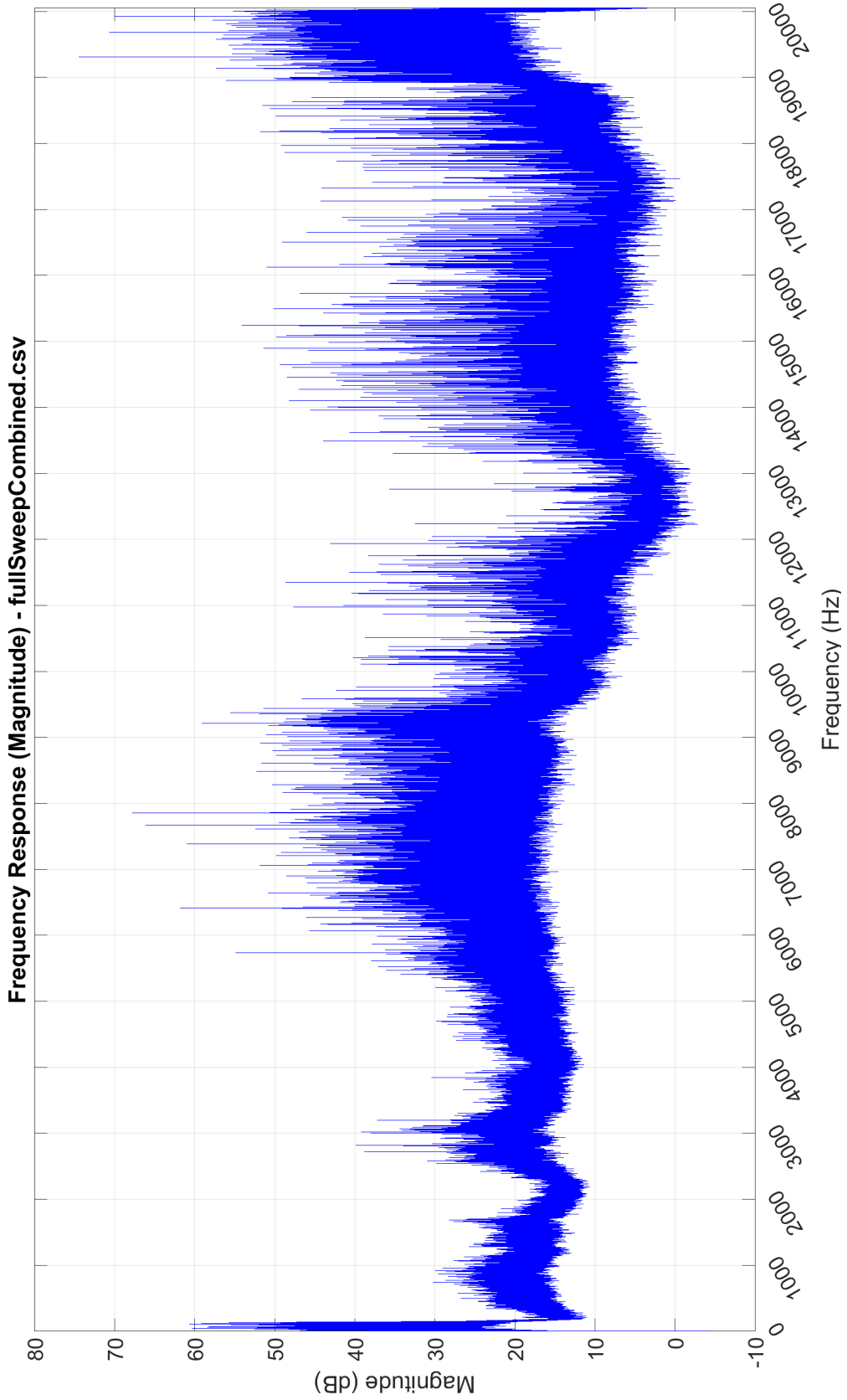


Figure K.6: Sine sweep from 20Hz to 20kHz frequency domain impulse response.

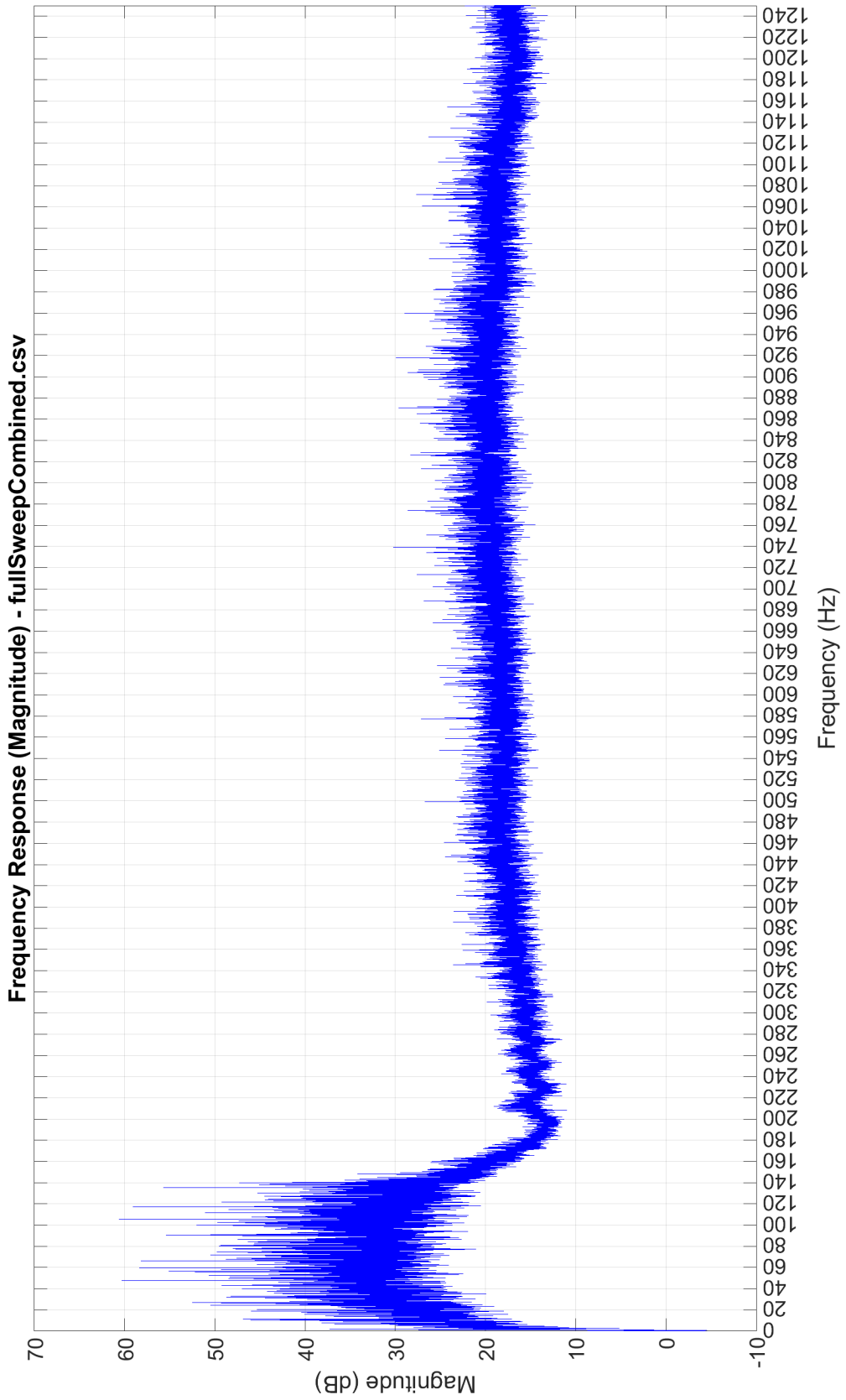


Figure K.7: Sine sweep from 20Hz to 20kHz frequency domain impulse response, with plot limited to 1250Hz.

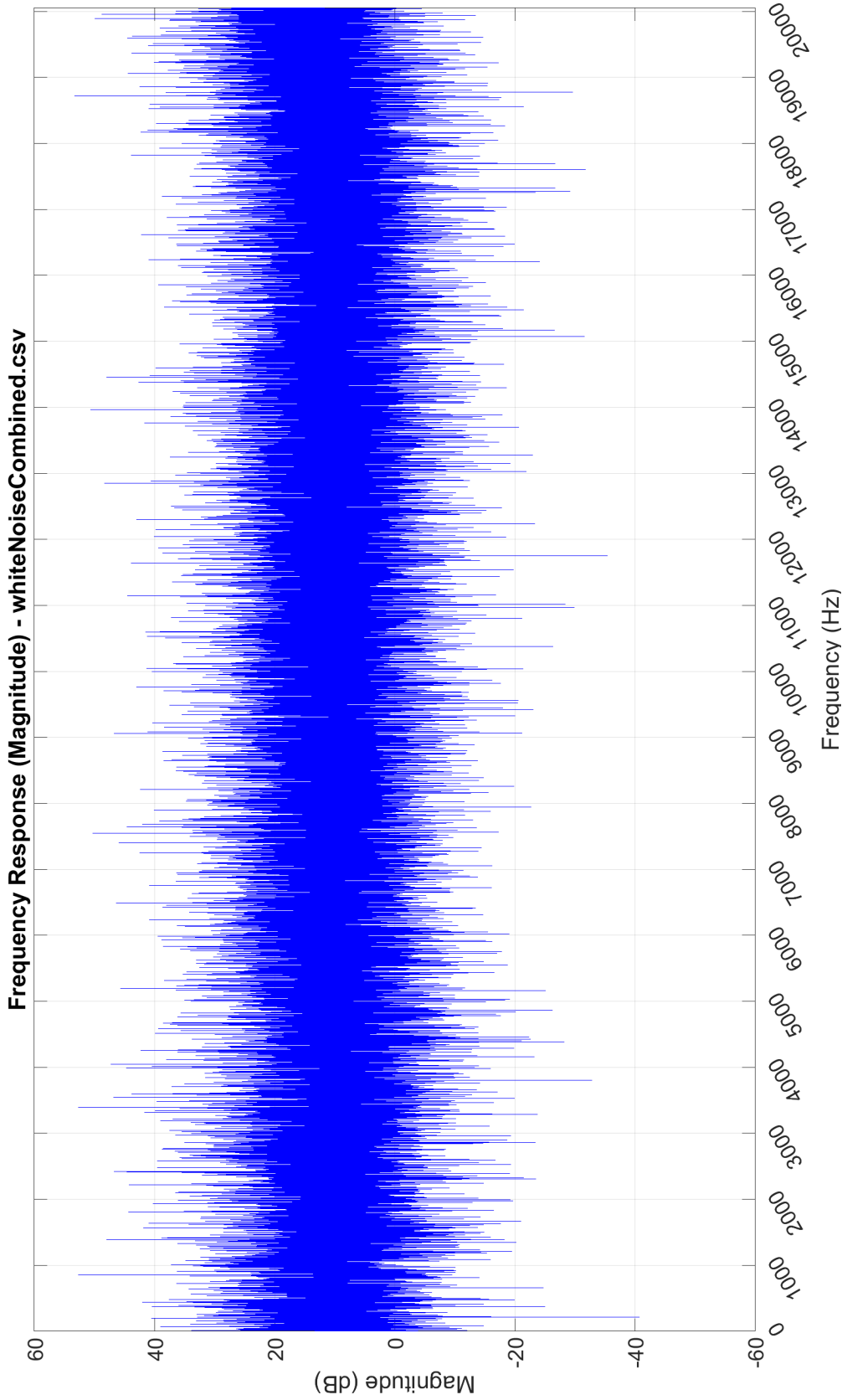


Figure K.8: White noise frequency domain impulse response.

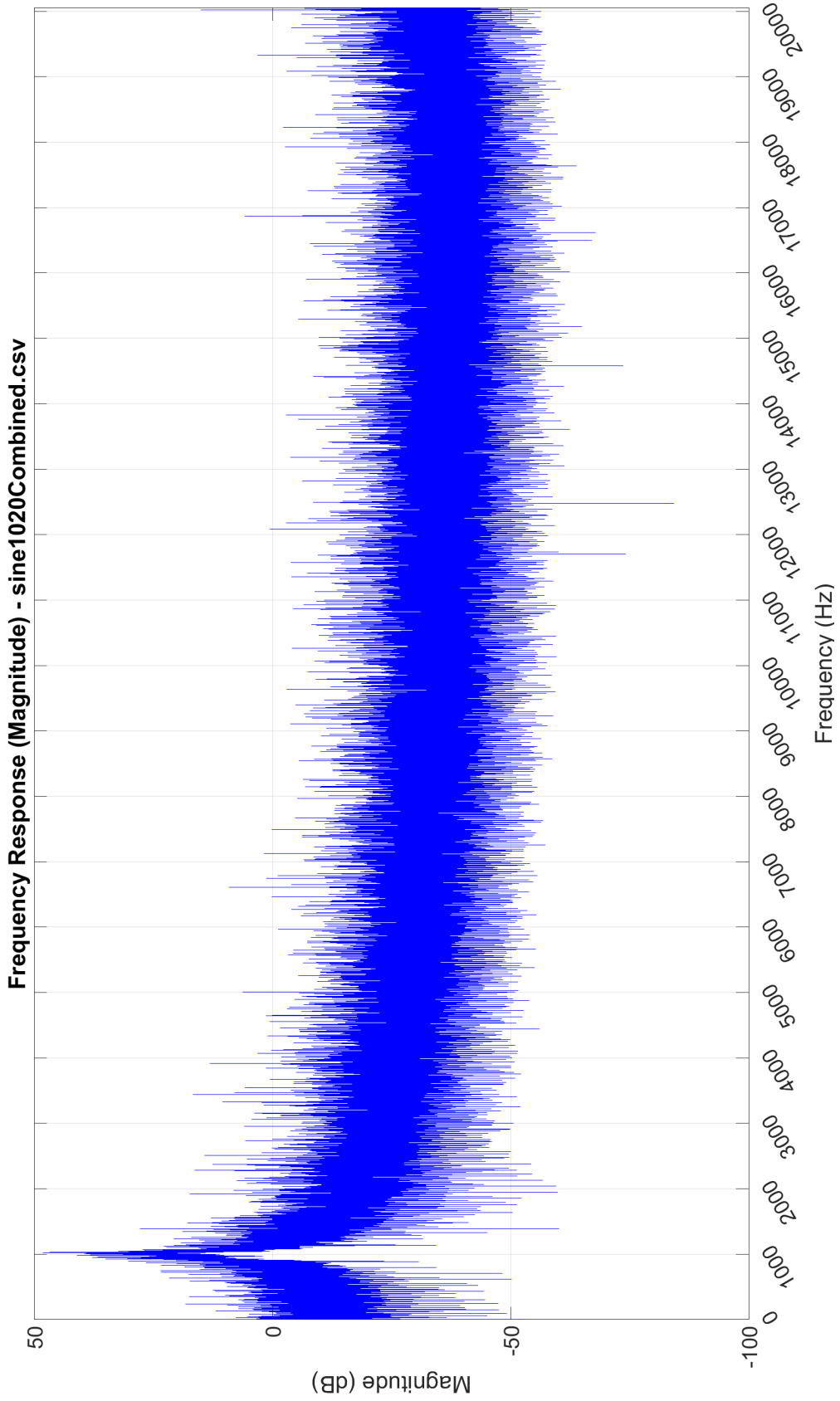


Figure K.9: 1020Hz puretone sinusoid frequency domain impulse response.

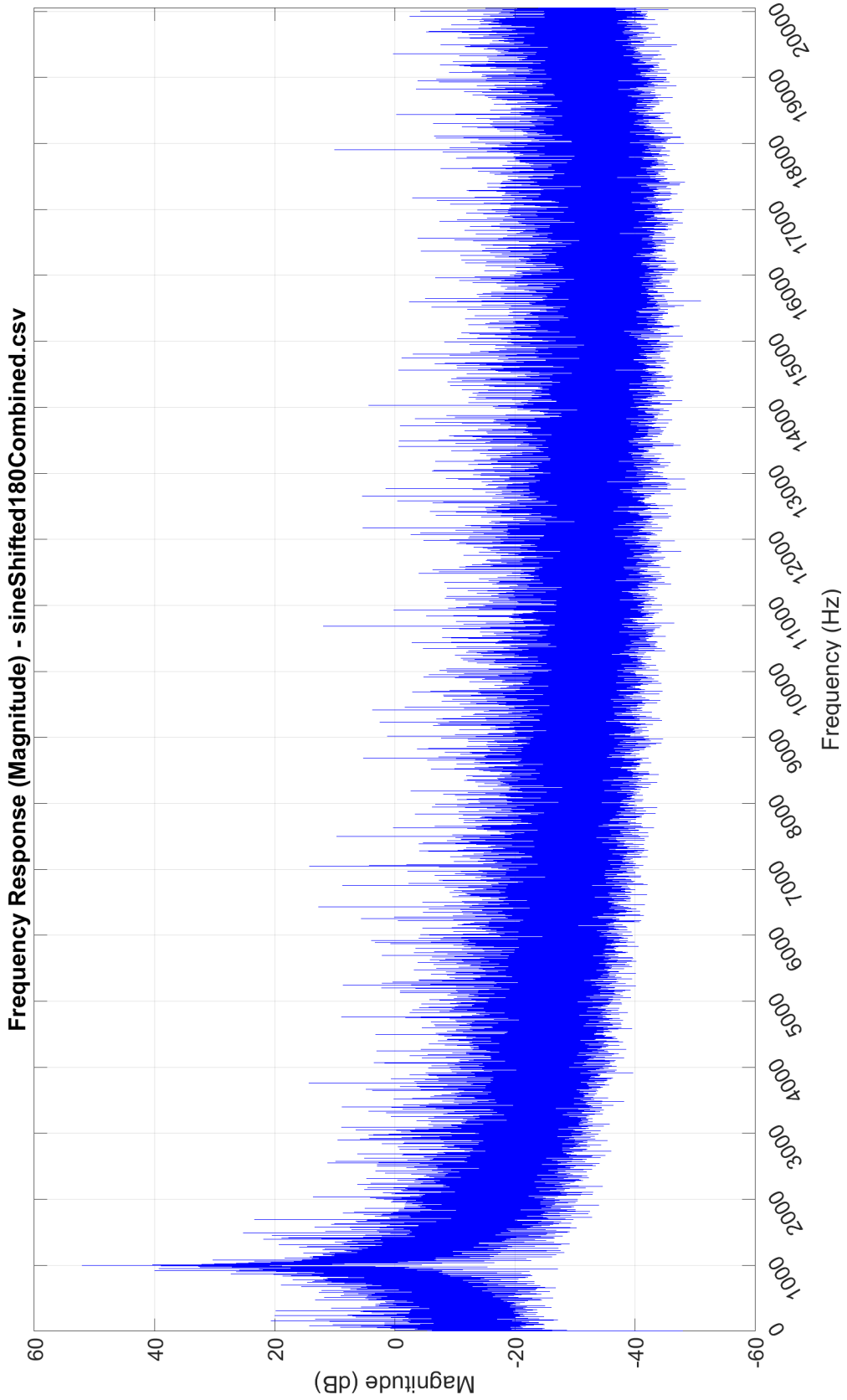


Figure K.10: 1kHz phase shifted sinusoid frequency domain impulse response.

L Frequency and Time Domain Plots 2 Microphones

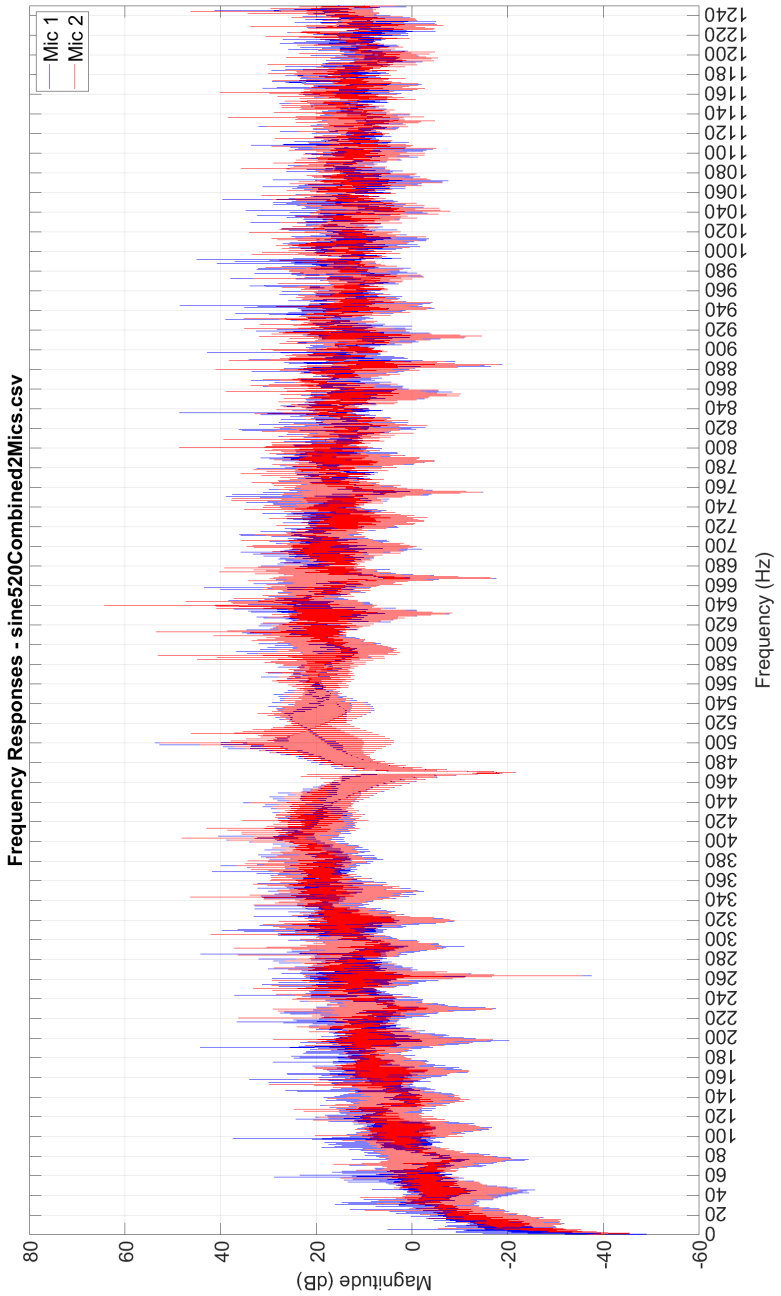


Figure L.1: Impulse response in the frequency domain of two microphones on the same I2S line at 520Hz.

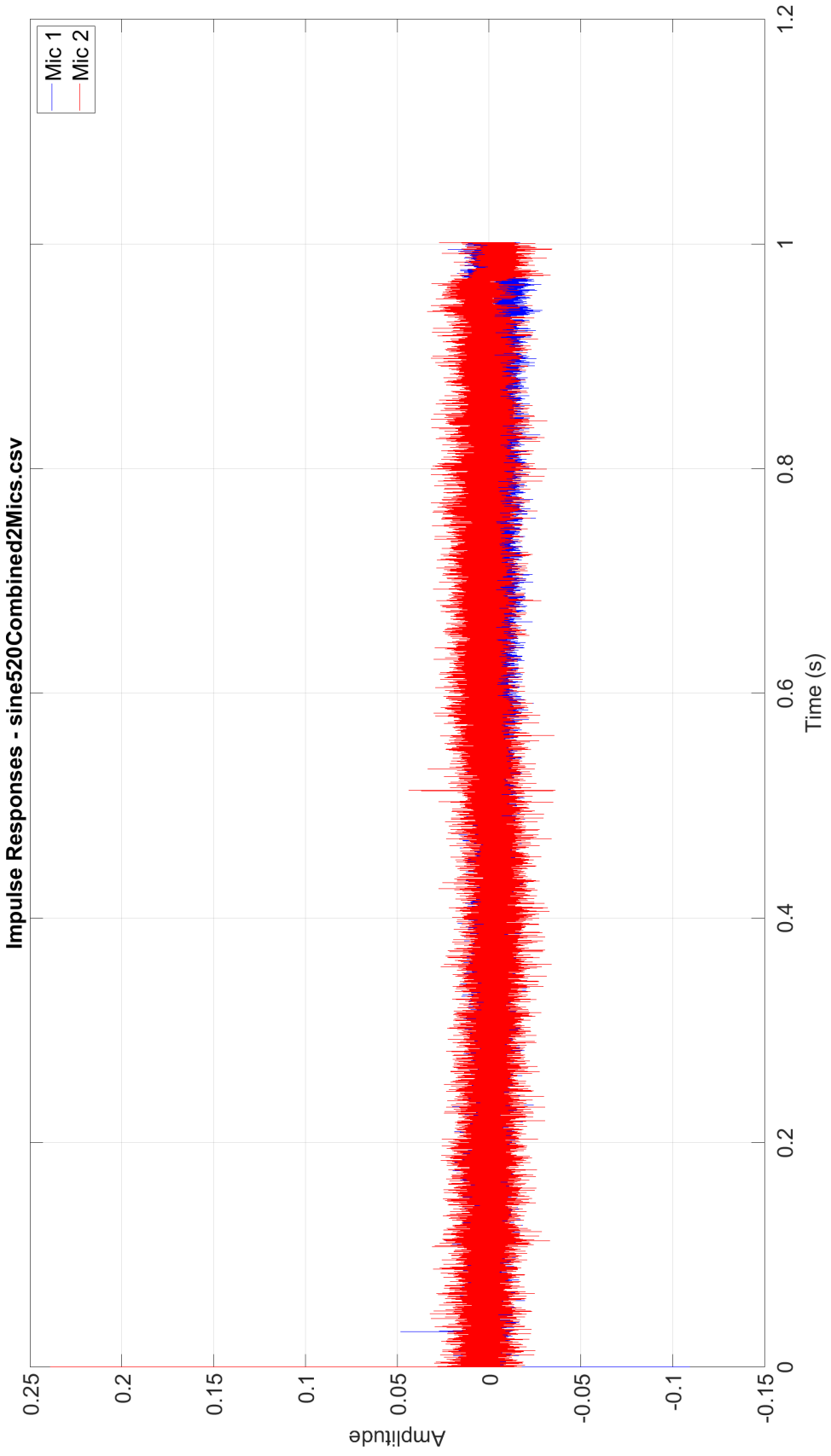


Figure L.2: Impulse response in the time domain of two microphones on the same I2S line at 520Hz.

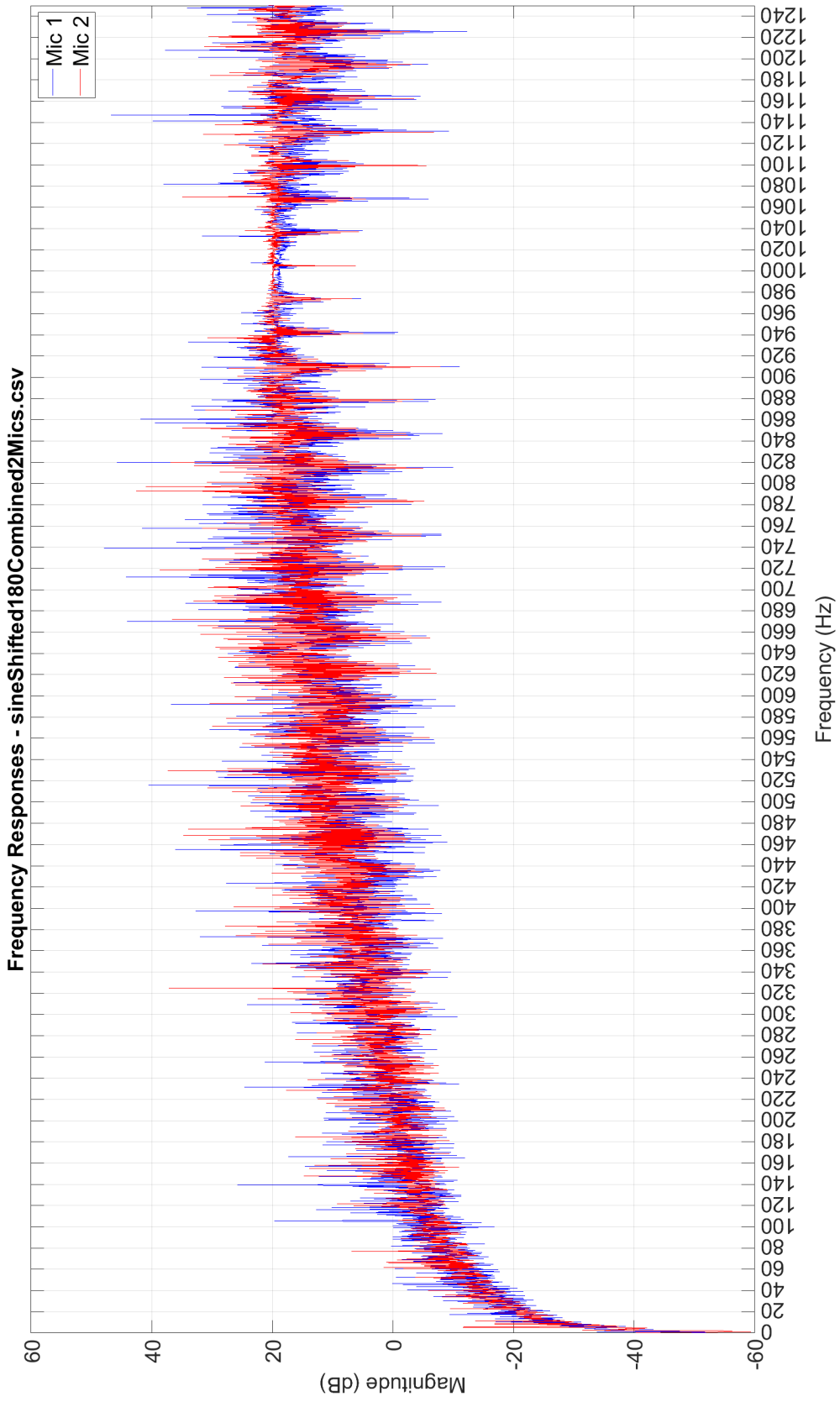


Figure L.3: Impulse response in the frequency domain of two microphones on the same I2S line at 1kHz phase shifted 180 degrees.

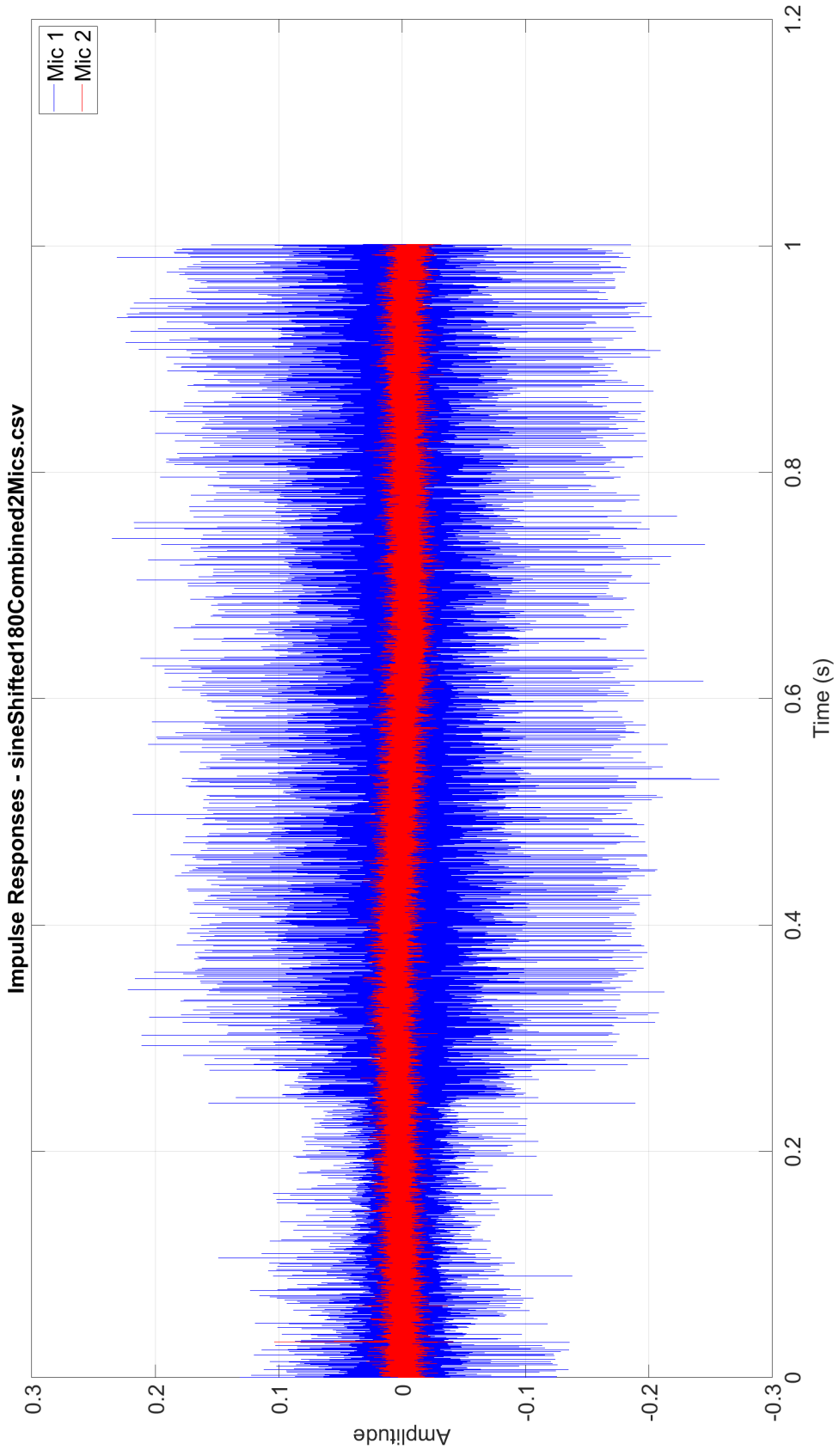


Figure L.4: Impulse response in the time domain of two microphones on the same I2S line at 1kHz phase shifted 180 degrees.

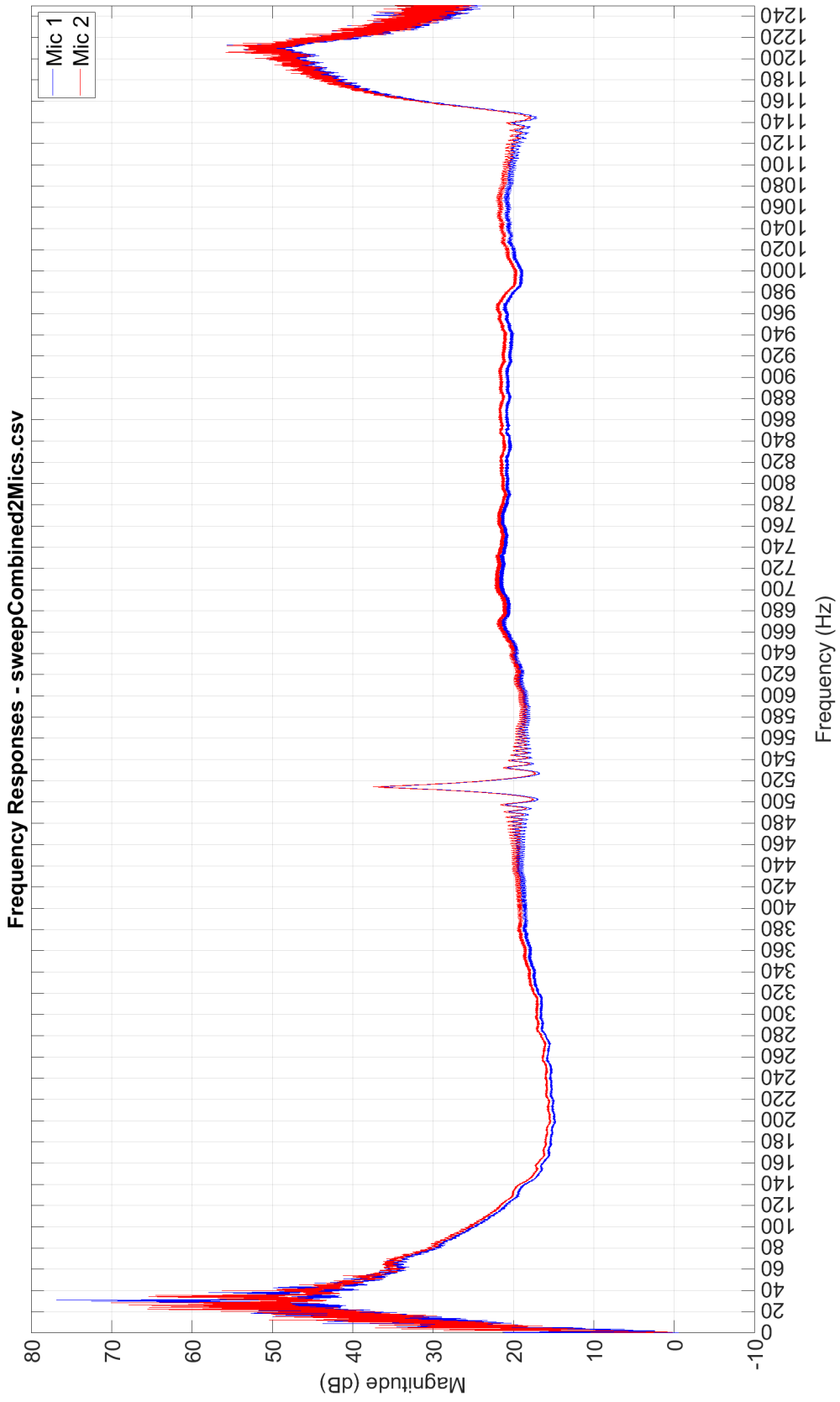


Figure L.5: Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz measured by two microphones on the same I2S line.

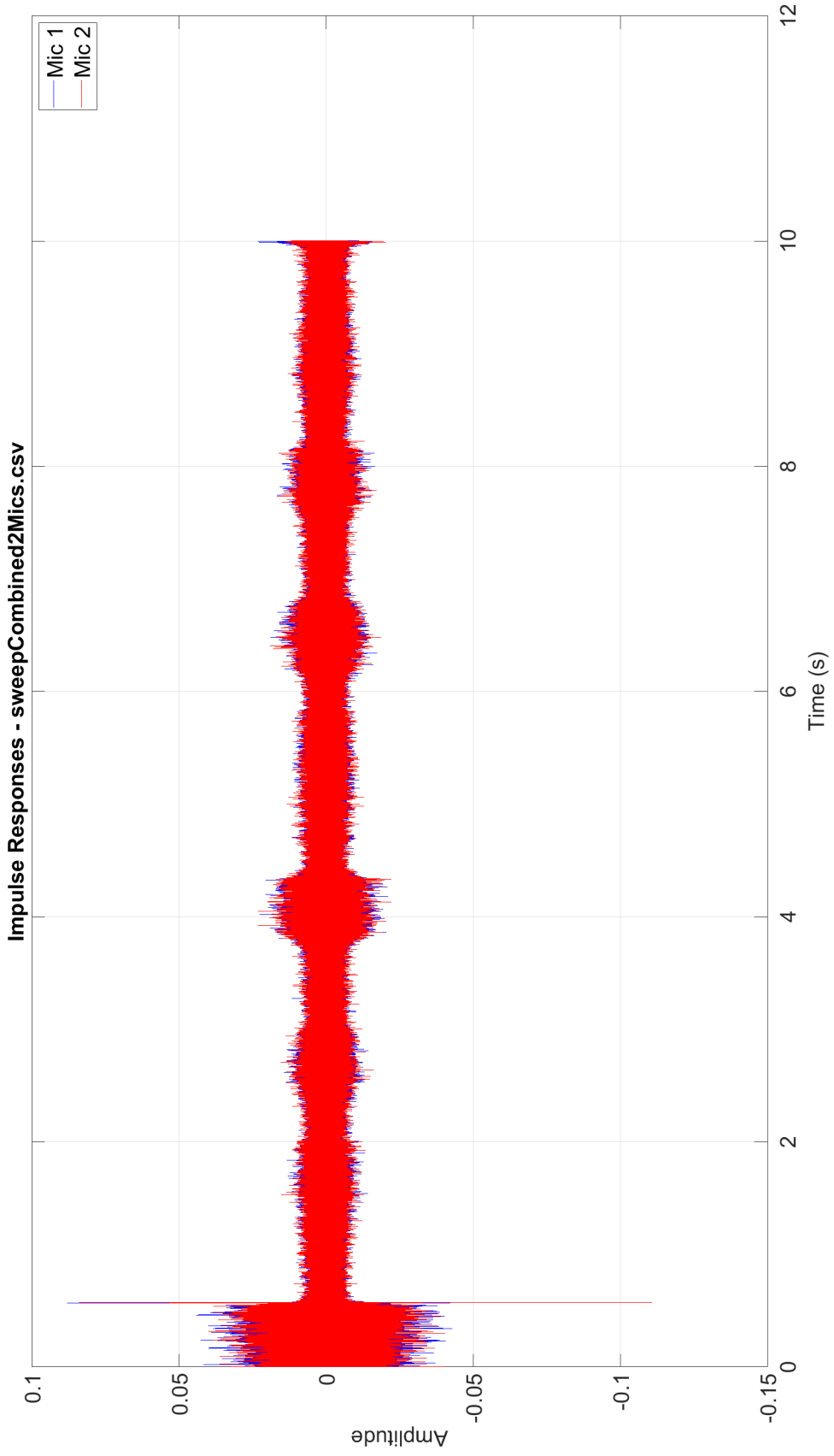


Figure L.6: Impulse response in the time domain of a sine sweep from 20Hz to 1220Hz measured by two microphones on the same I2S line.

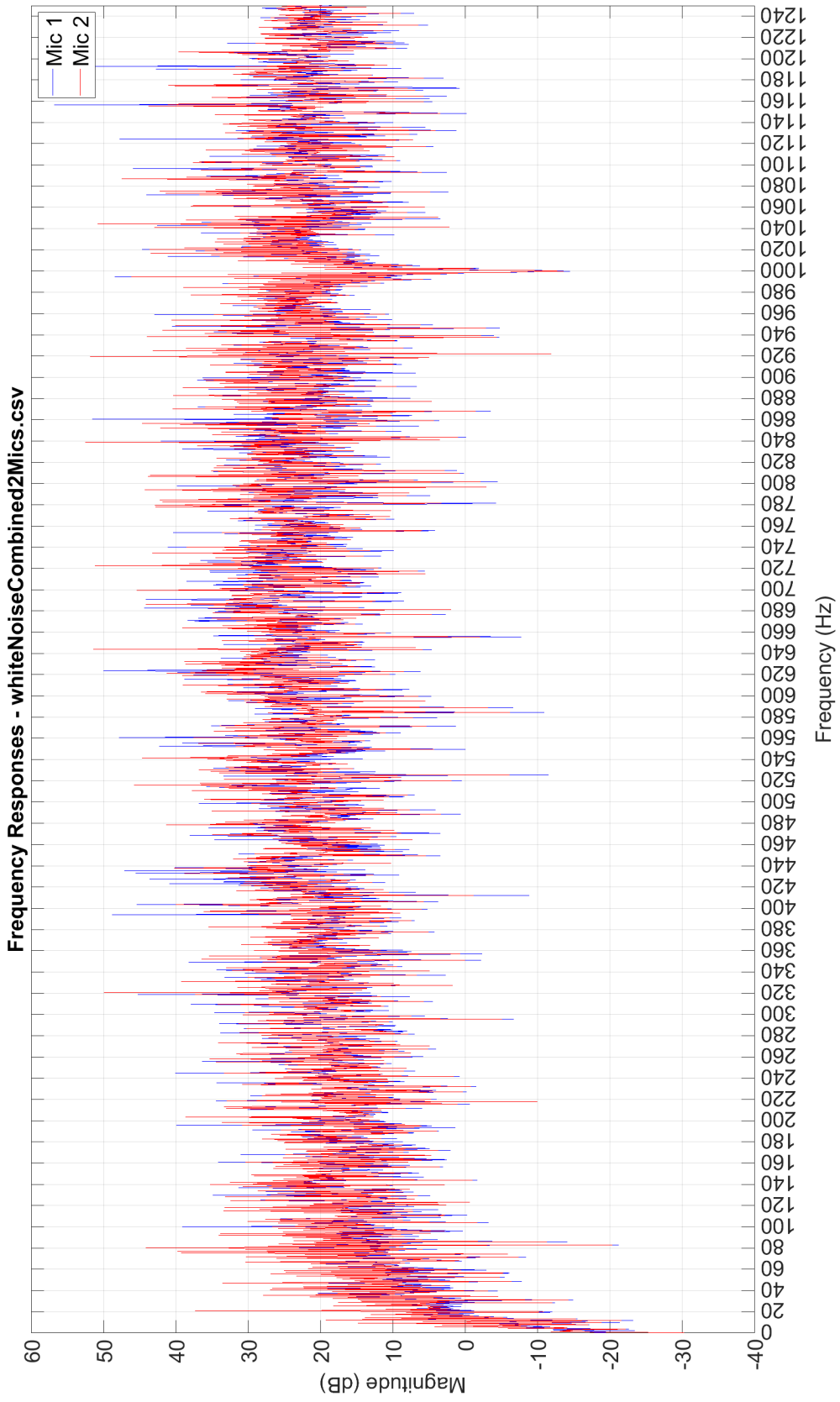


Figure L.7: Impulse response in the frequency domain of white noise measured by two microphones on the same I2S line.

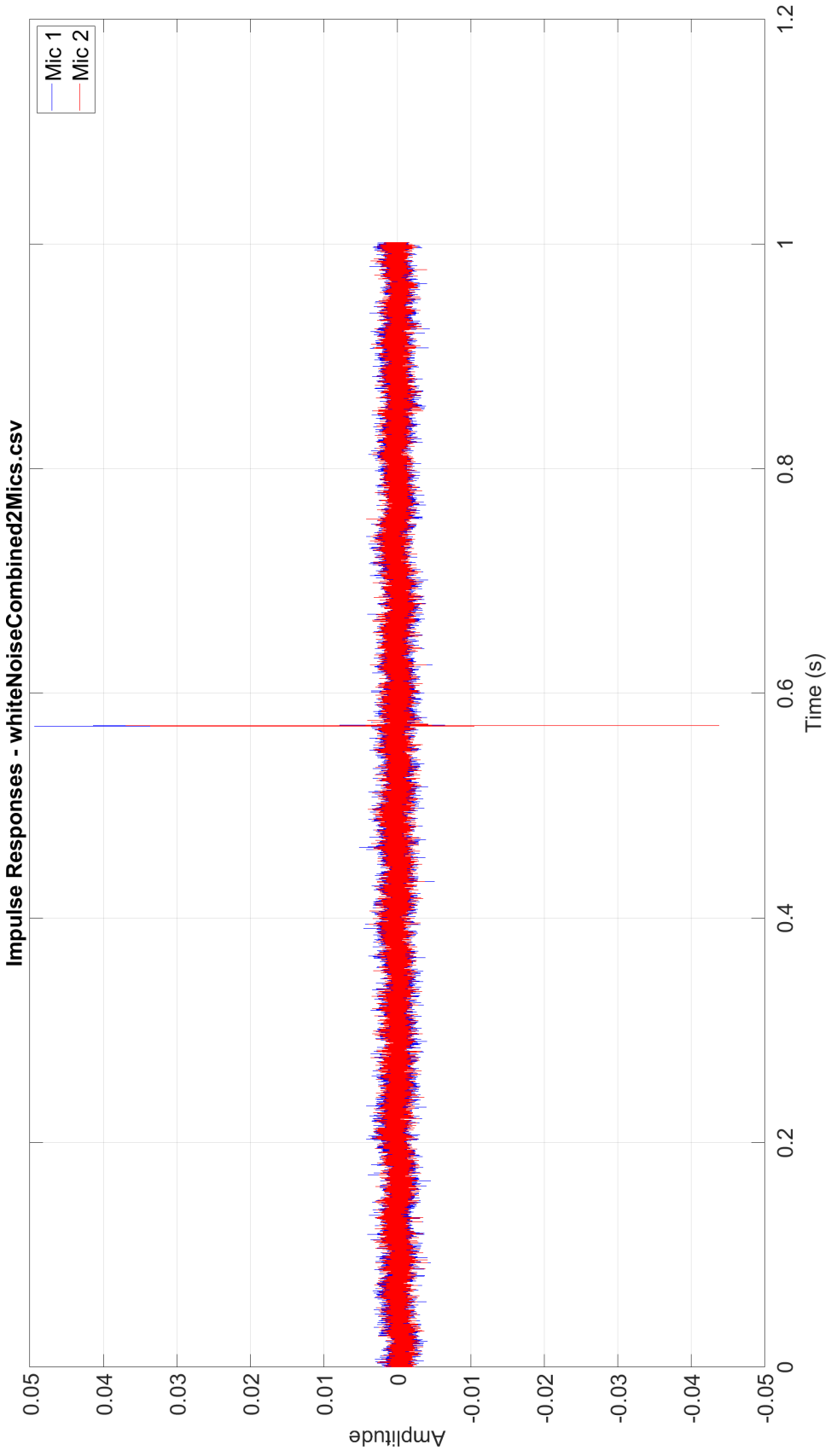


Figure L.8: Impulse response in the time domain of white noise measured by two microphones on the same I2S line.

M | Frequency and Time Domain Plots 6 Microphones

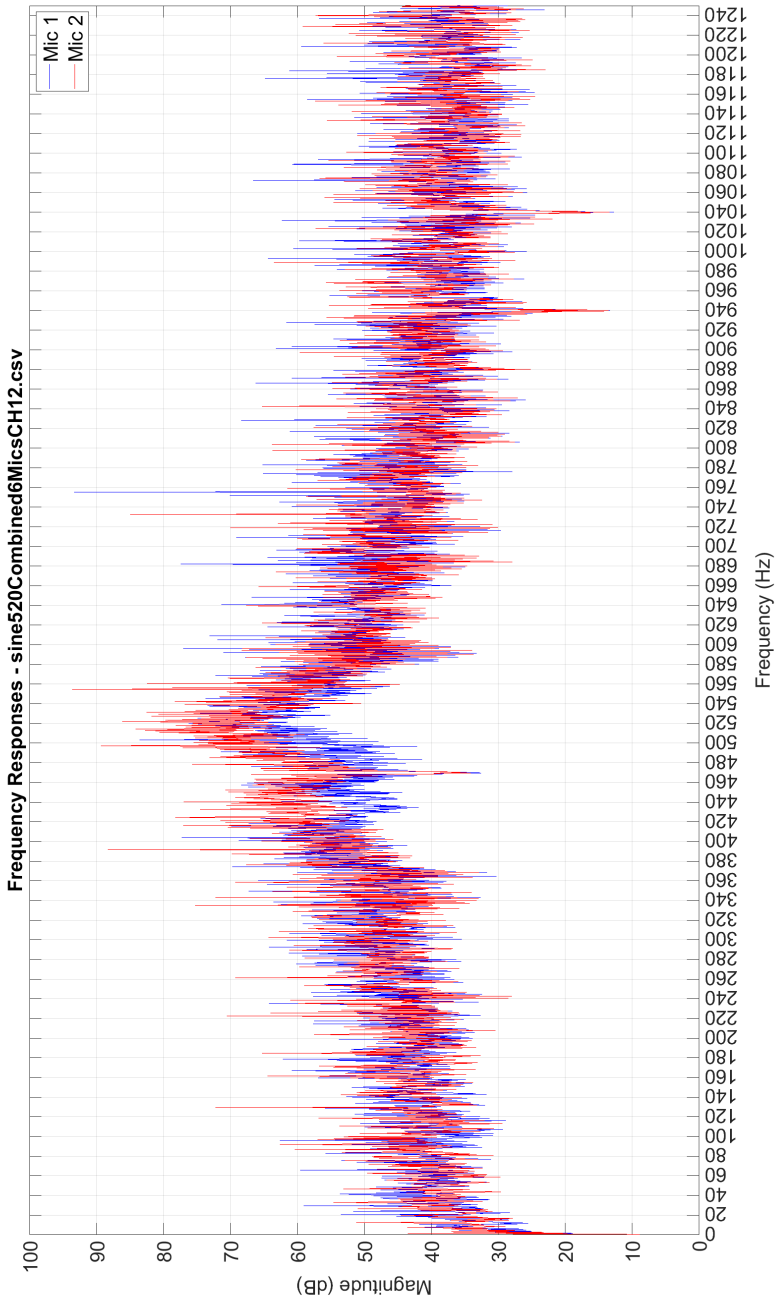


Figure M.1: Impulse response in the frequency domain of channel 1/2 at 520Hz.

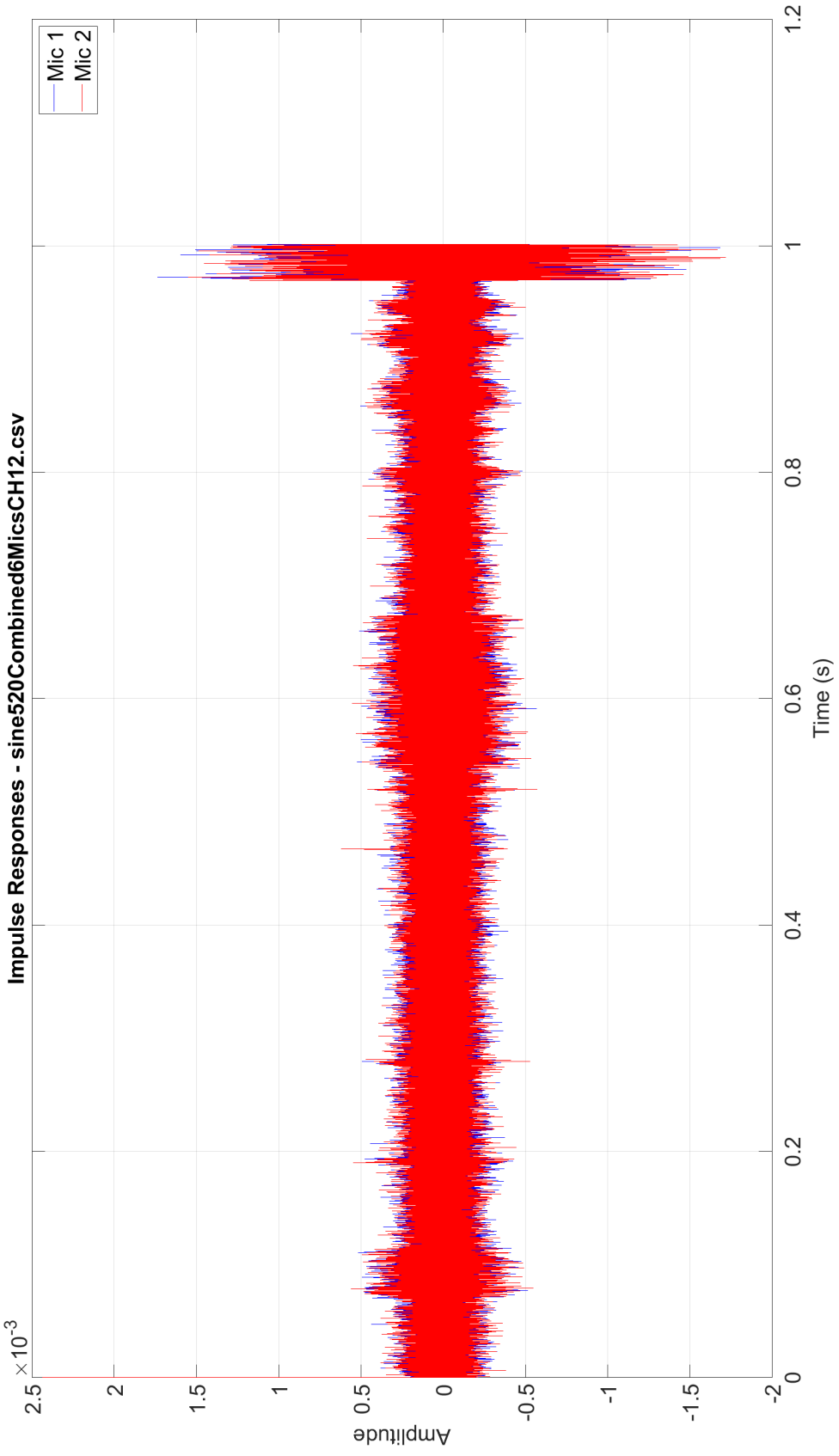


Figure M.2: Impulse response in the time domain of channel 1/2 at 520Hz.

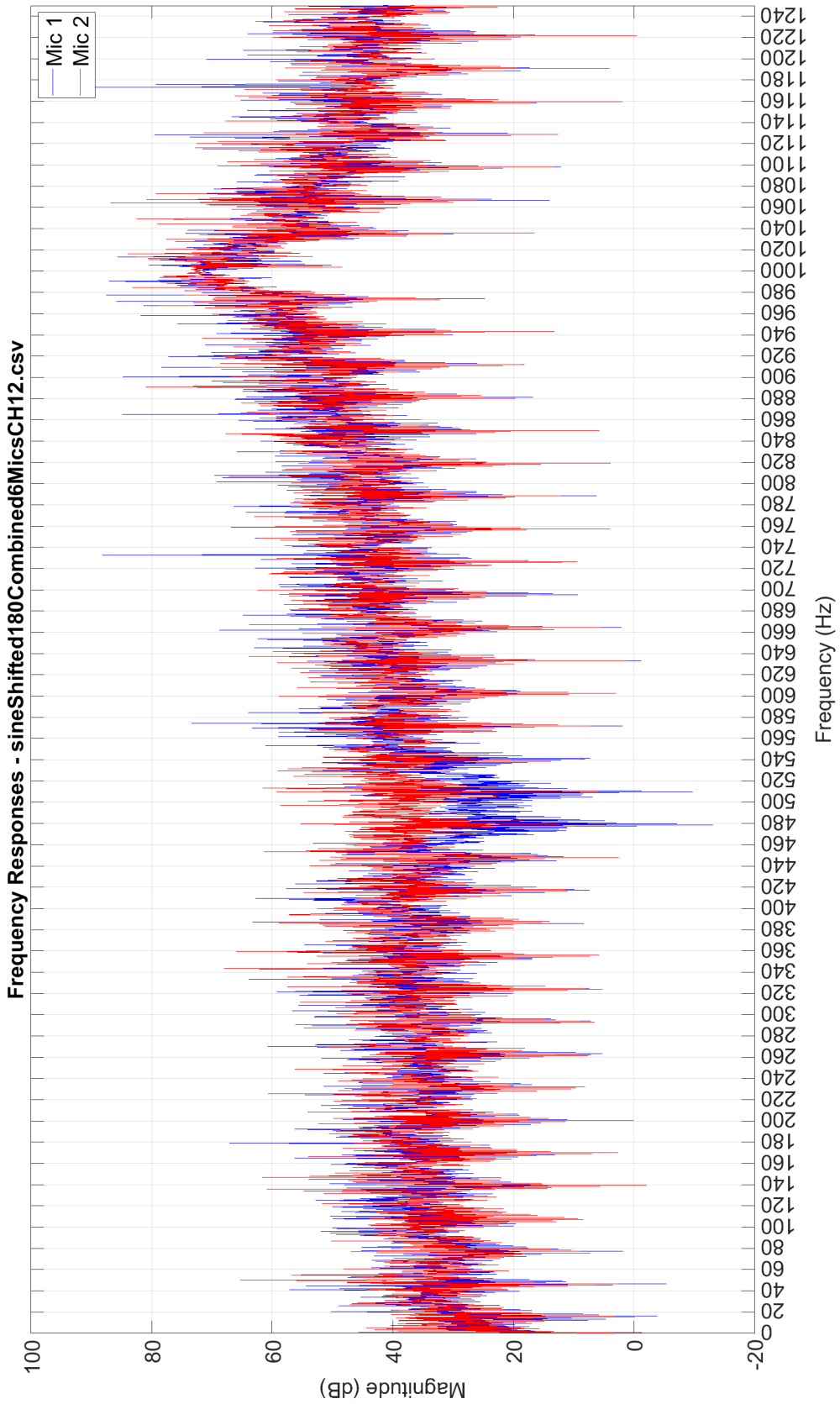


Figure M.3: Impulse response in the frequency domain of channel 1/2 at 1kHz phase shifted 180 degrees.

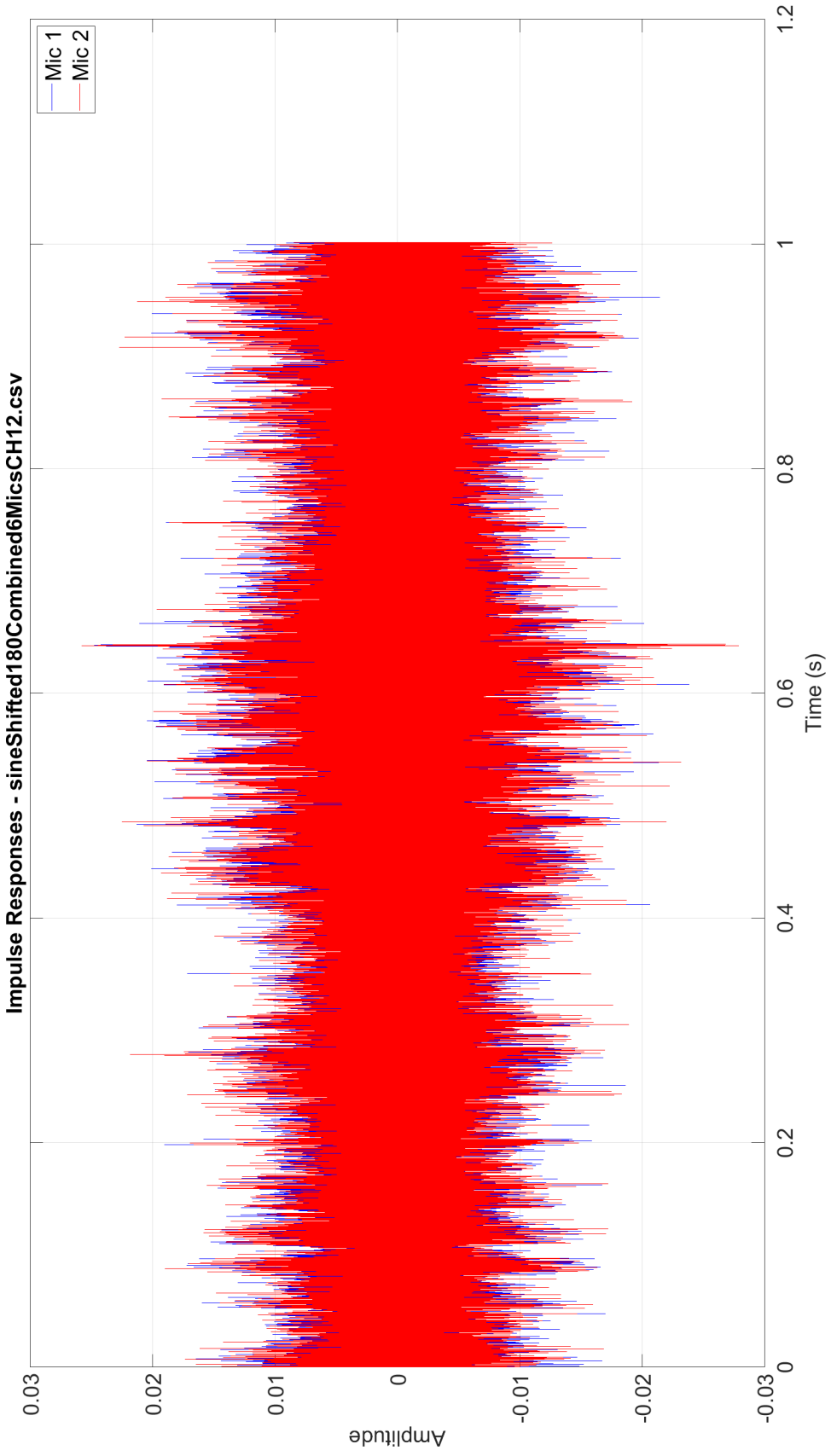


Figure M.4: Impulse response in the time domain of channel 1/2 at 1kHz phase shifted 180 degrees.

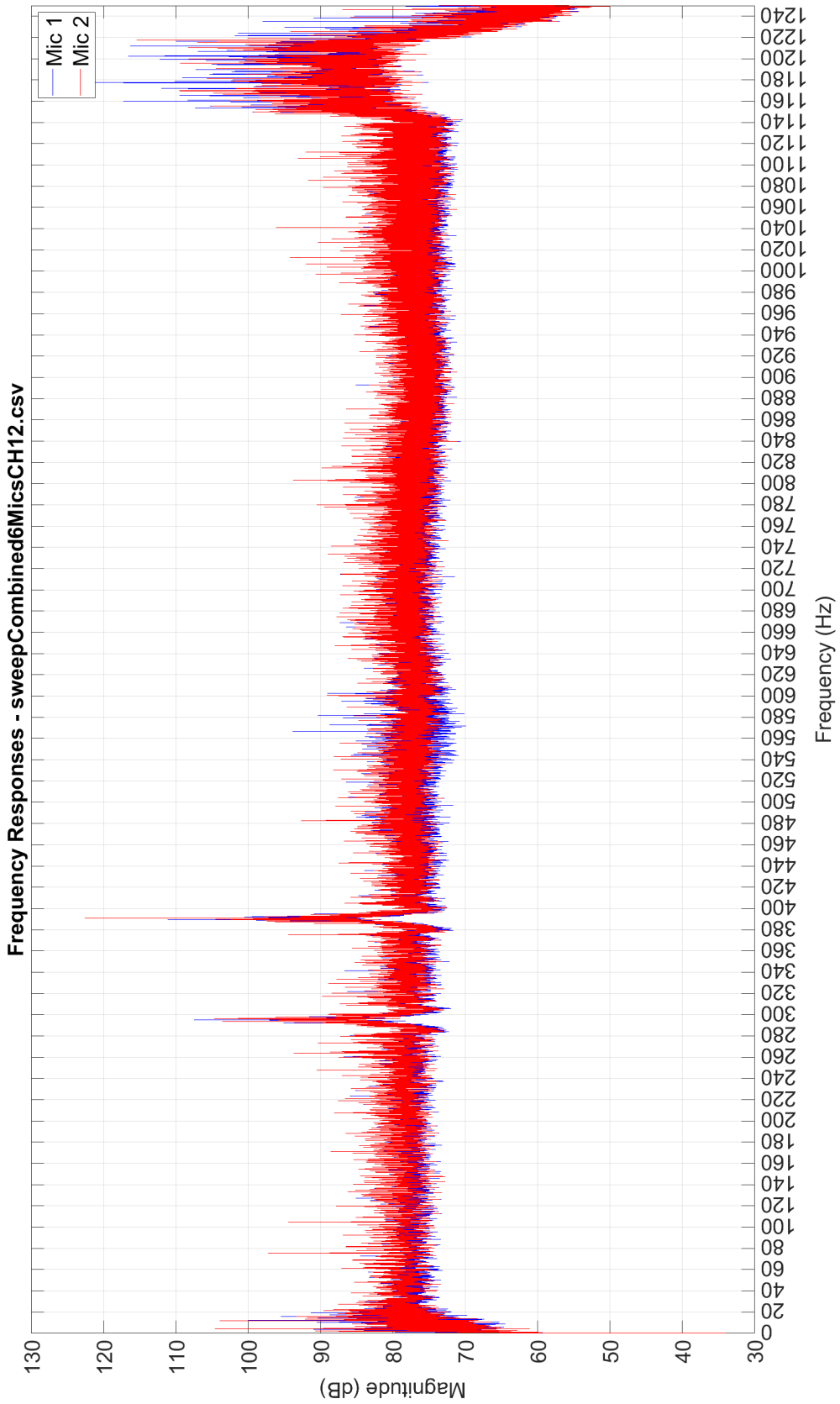


Figure M.5: Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz on channel 1/2.

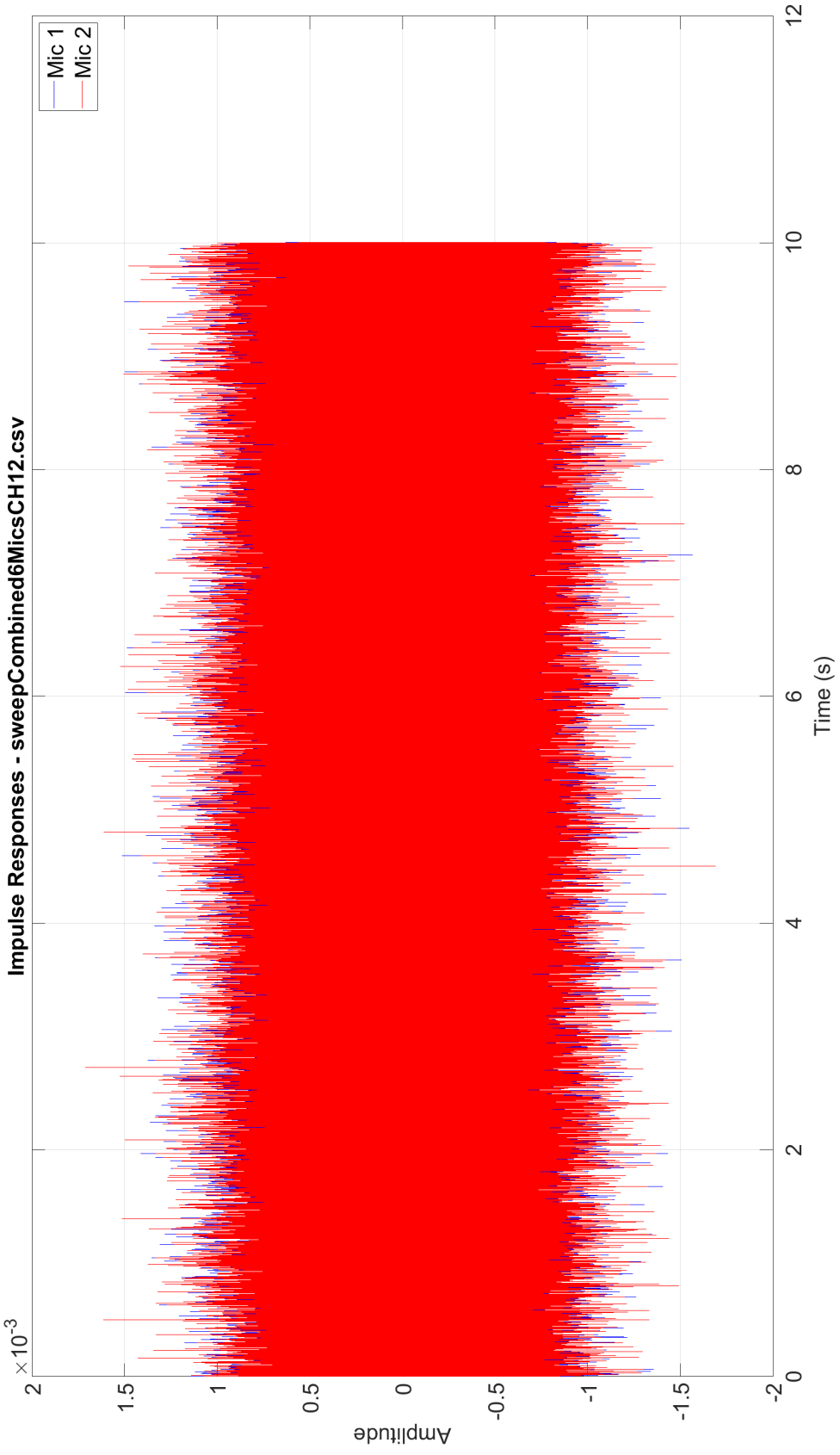


Figure M.6: Impulse response in the time domain of a sine sweep from 20Hz to 1220Hz on channel 1/2.

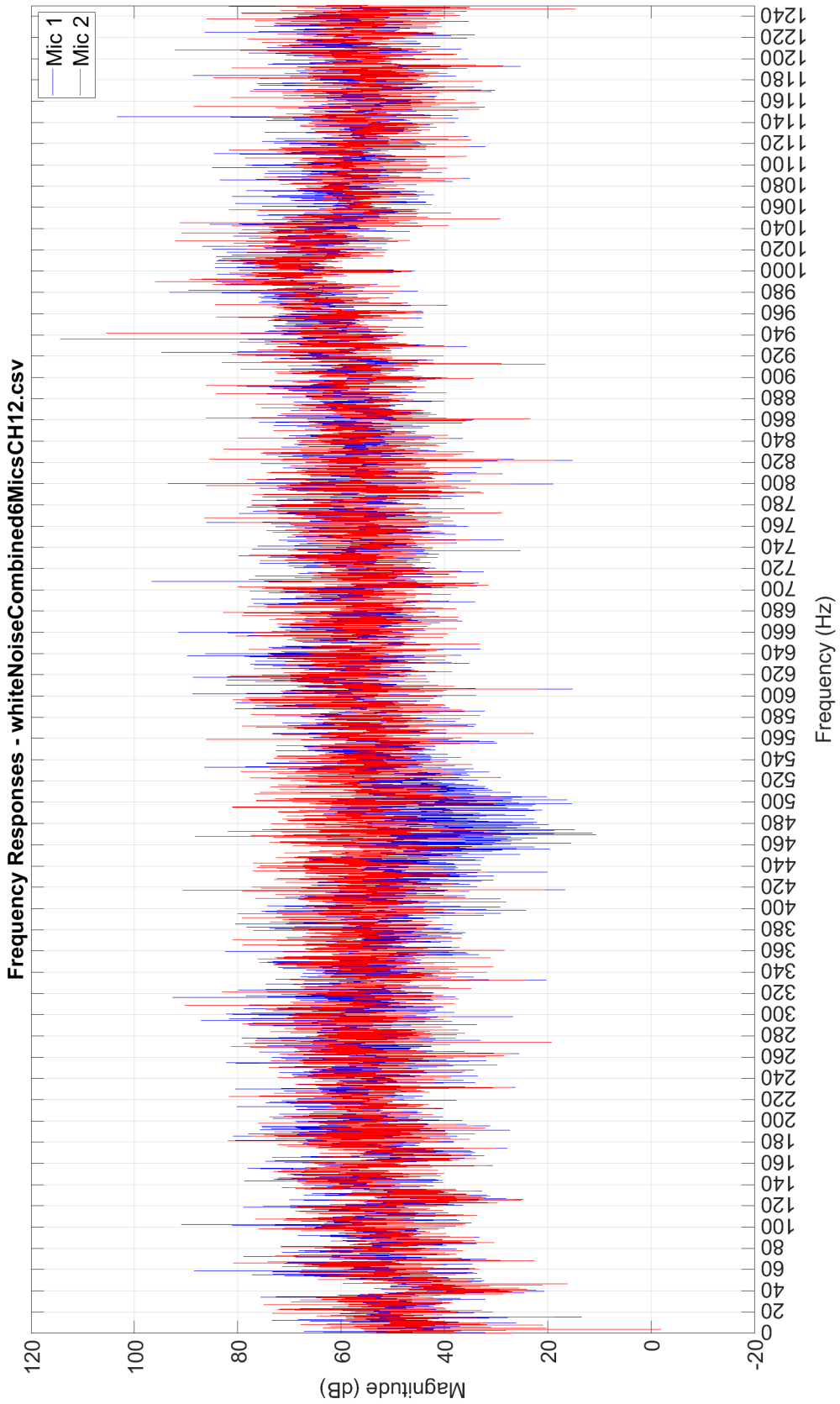


Figure M.7: Impulse response in the frequency domain of white noise on channel 1/2.

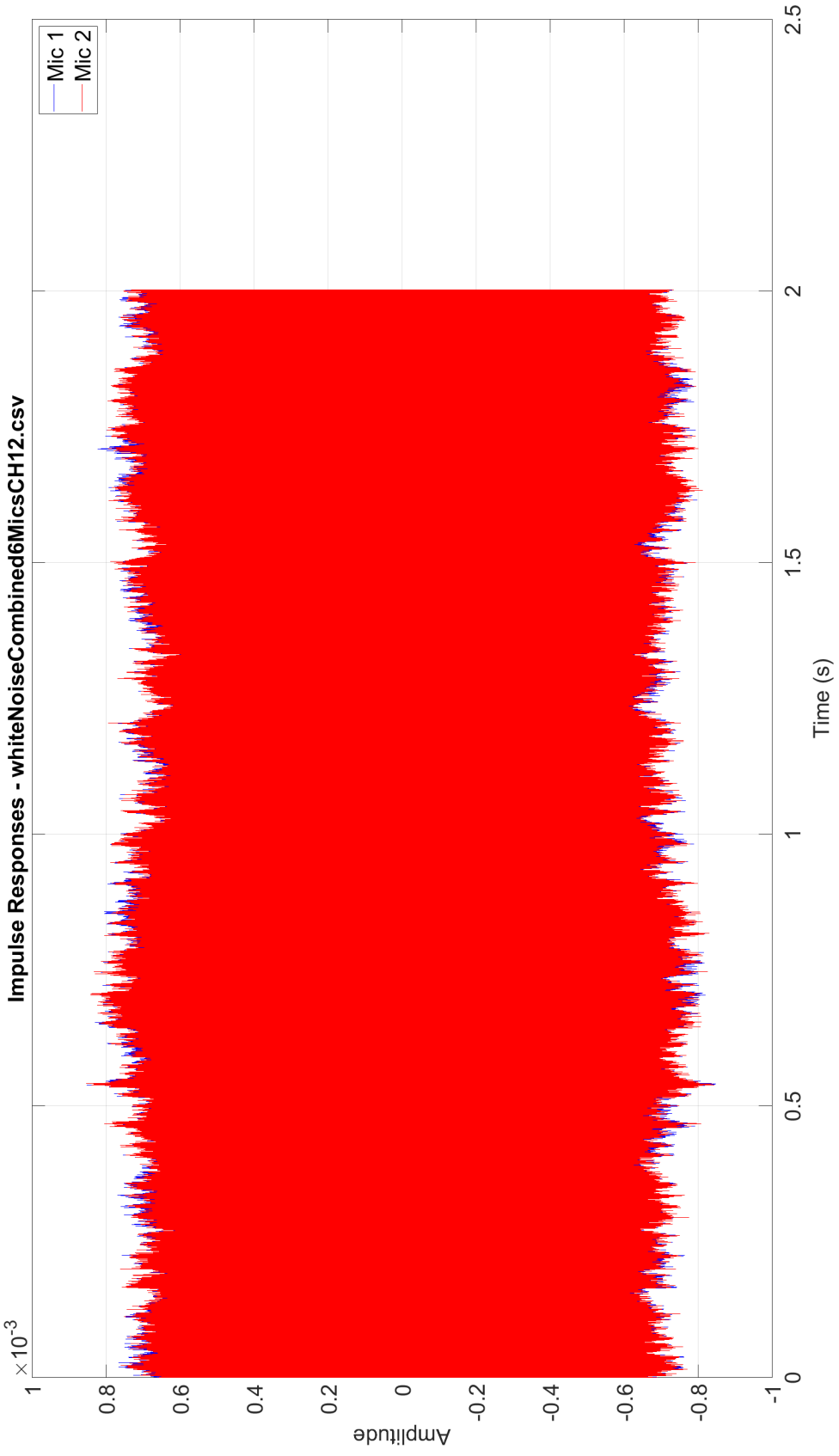


Figure M.8: Impulse response in the time domain of white noise on channel 1/2.

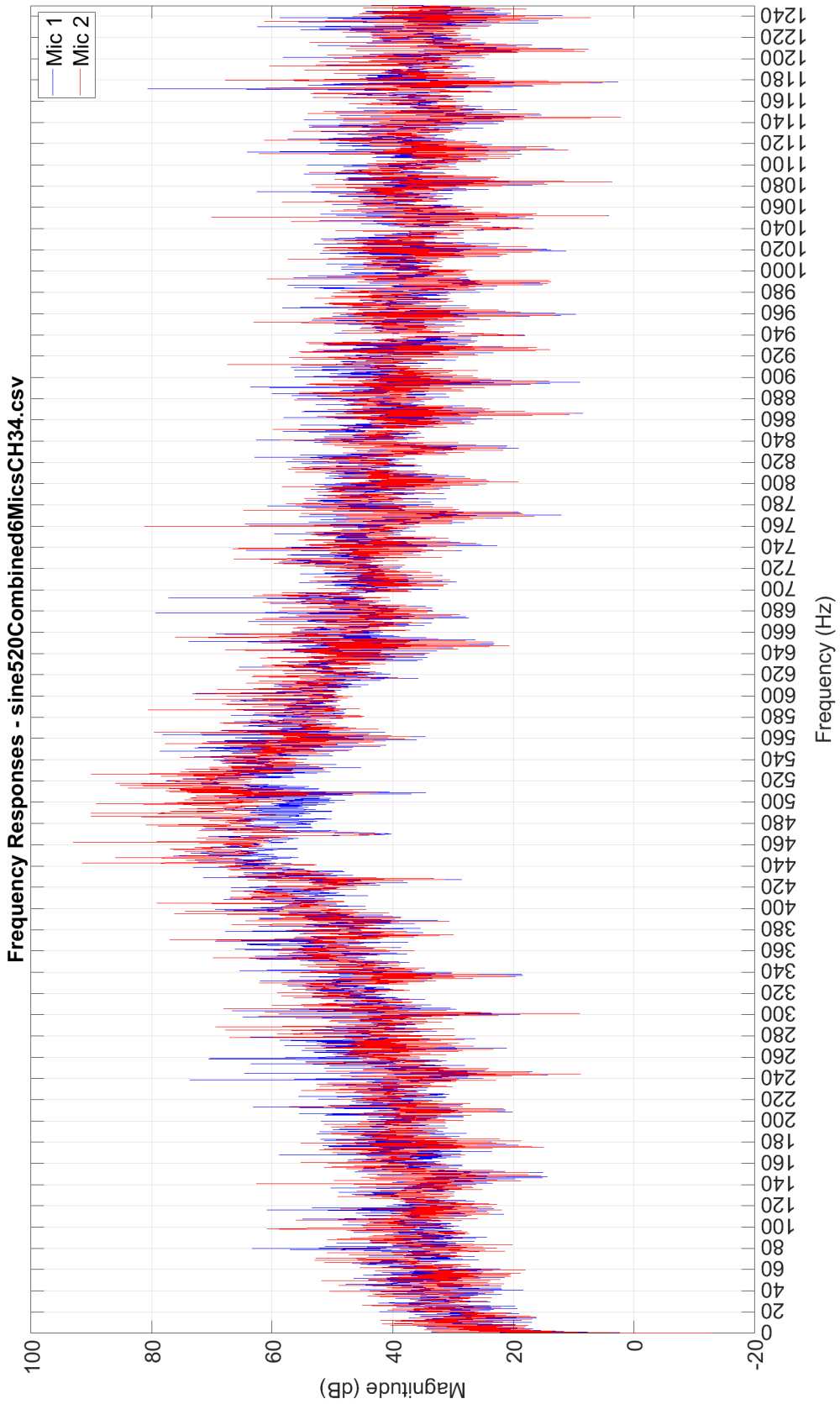


Figure M.9: Impulse response in the frequency domain of channel 3/4 at 520Hz.

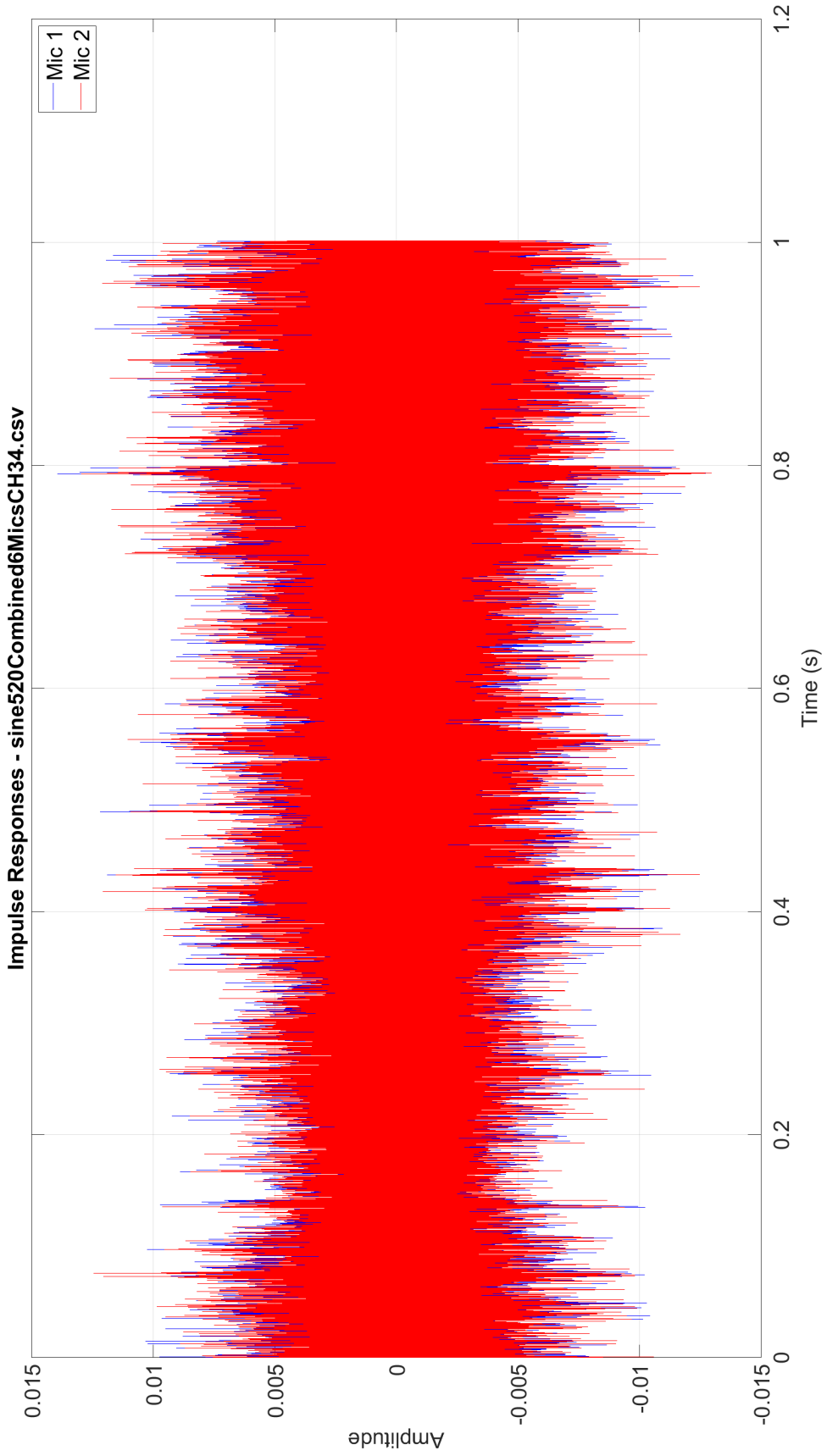


Figure M.10: Impulse response in the time domain of channel 3/4 at 520Hz.

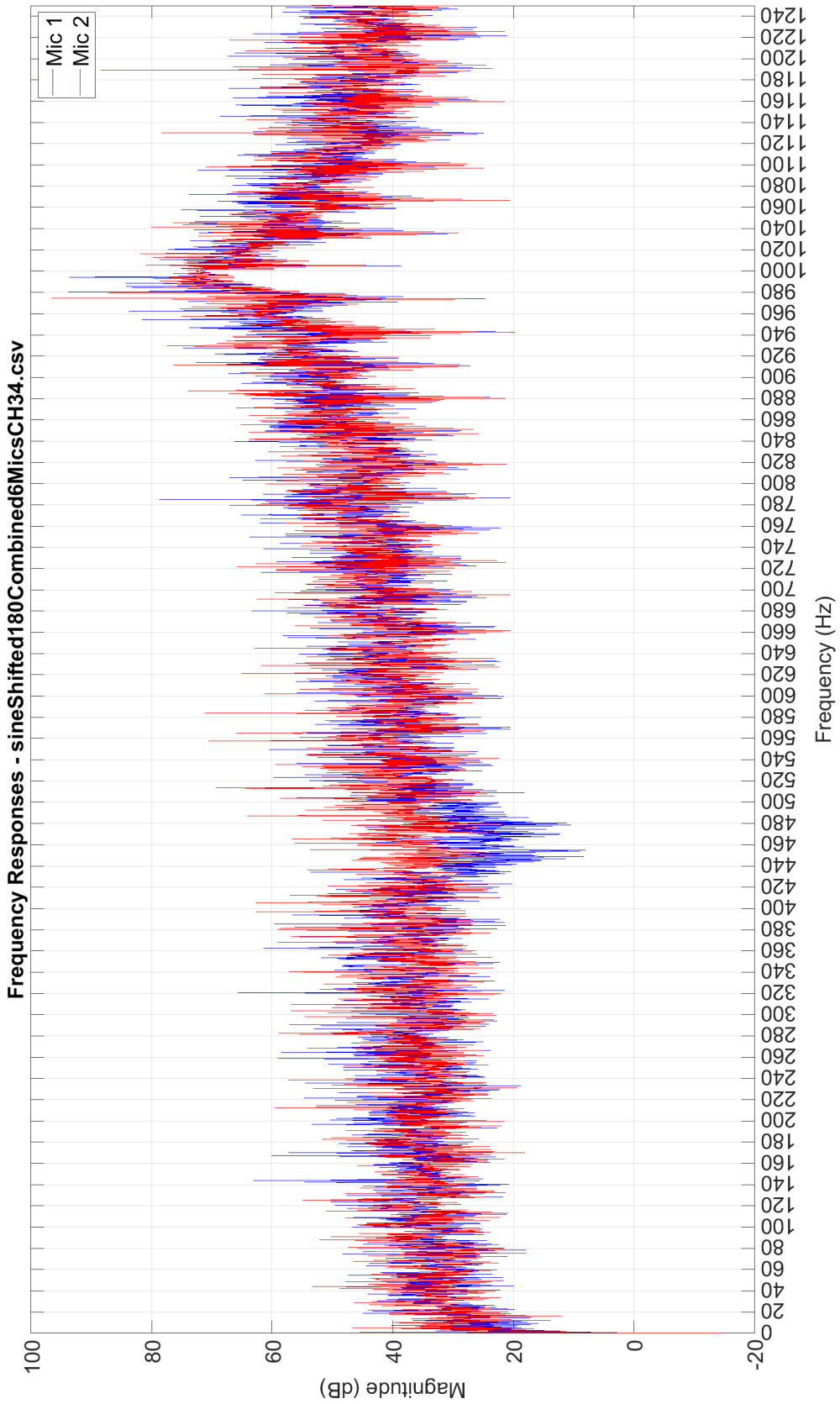


Figure M.11: Impulse response in the frequency domain of channel 3/4 at 1kHz phase shifted 180 degrees.

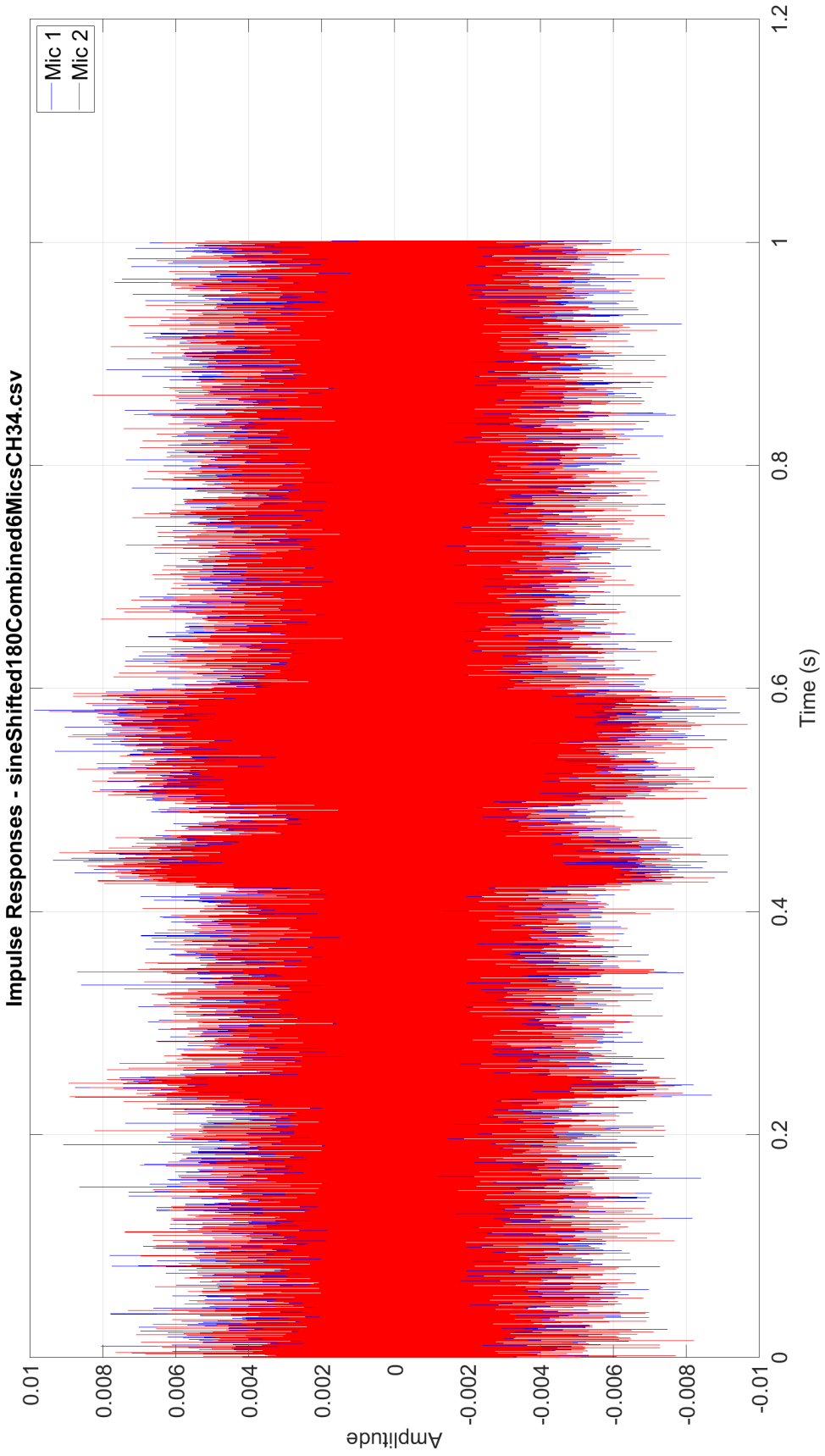


Figure M.12: Impulse response in the time domain of channel 3/4 at 1kHz phase shifted 180 degrees.

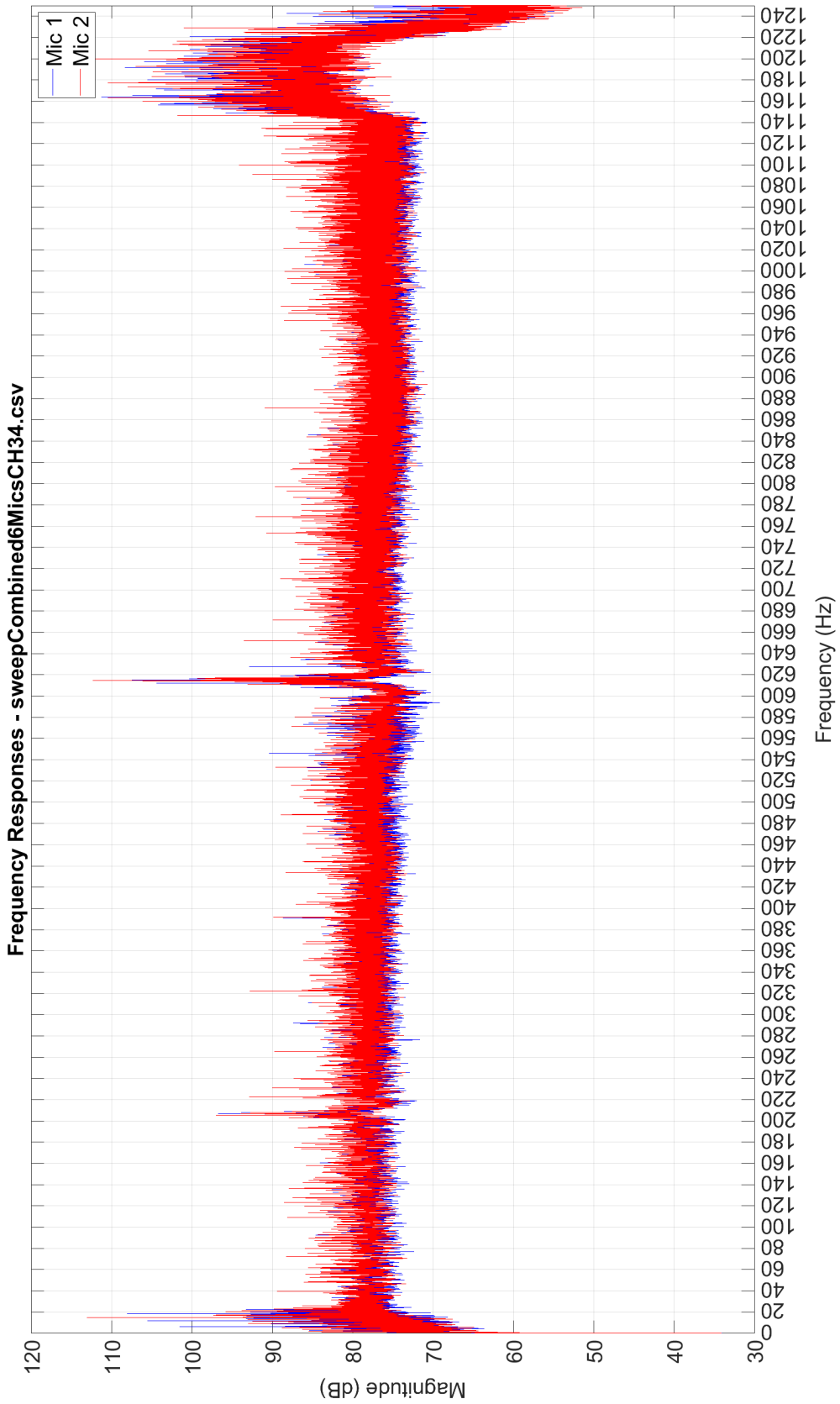


Figure M.13: Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz on channel 3/4.

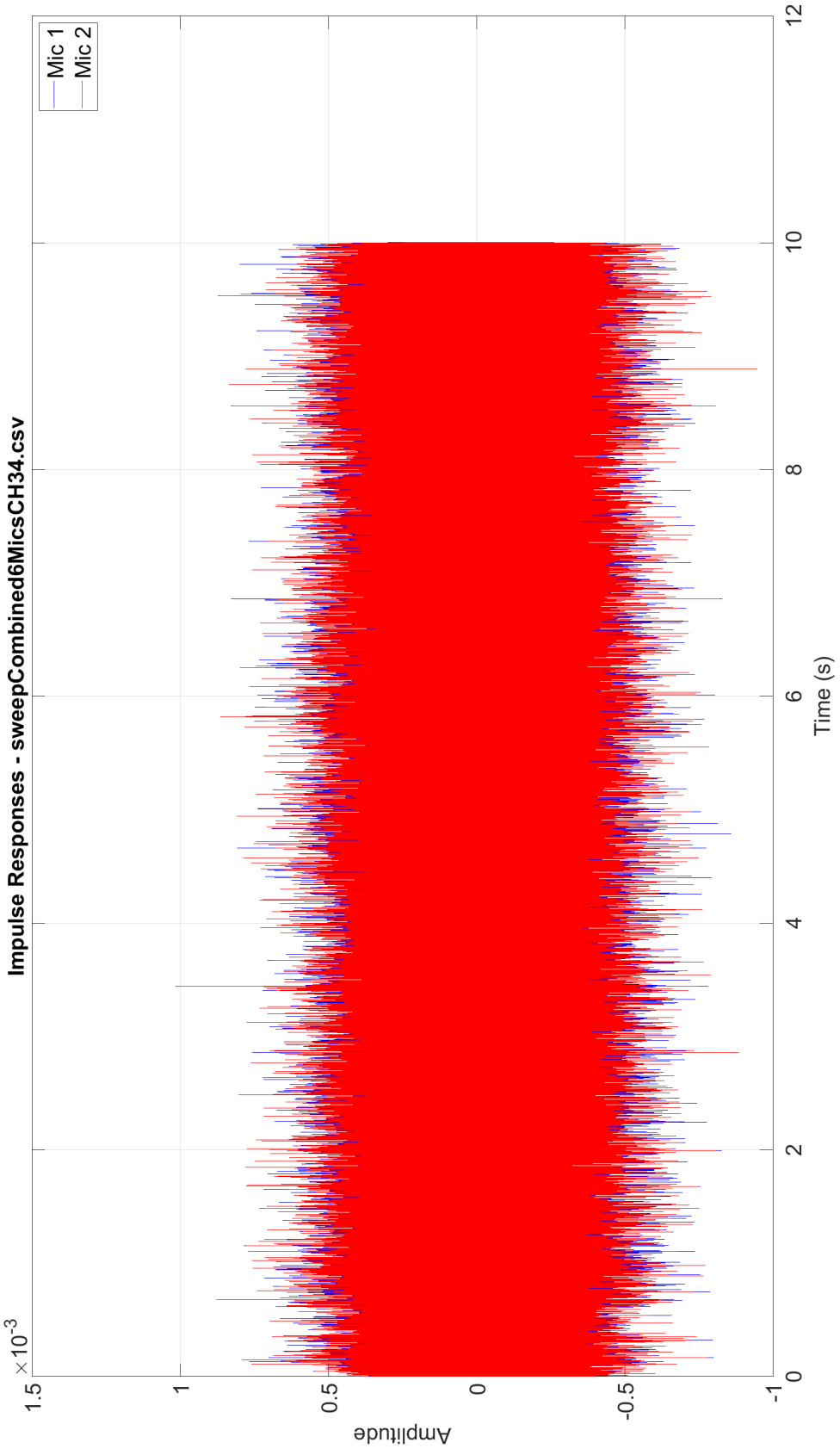


Figure M.14: Impulse response in the time domain of a sine sweep from 20Hz to 1220Hz on channel 3/4.

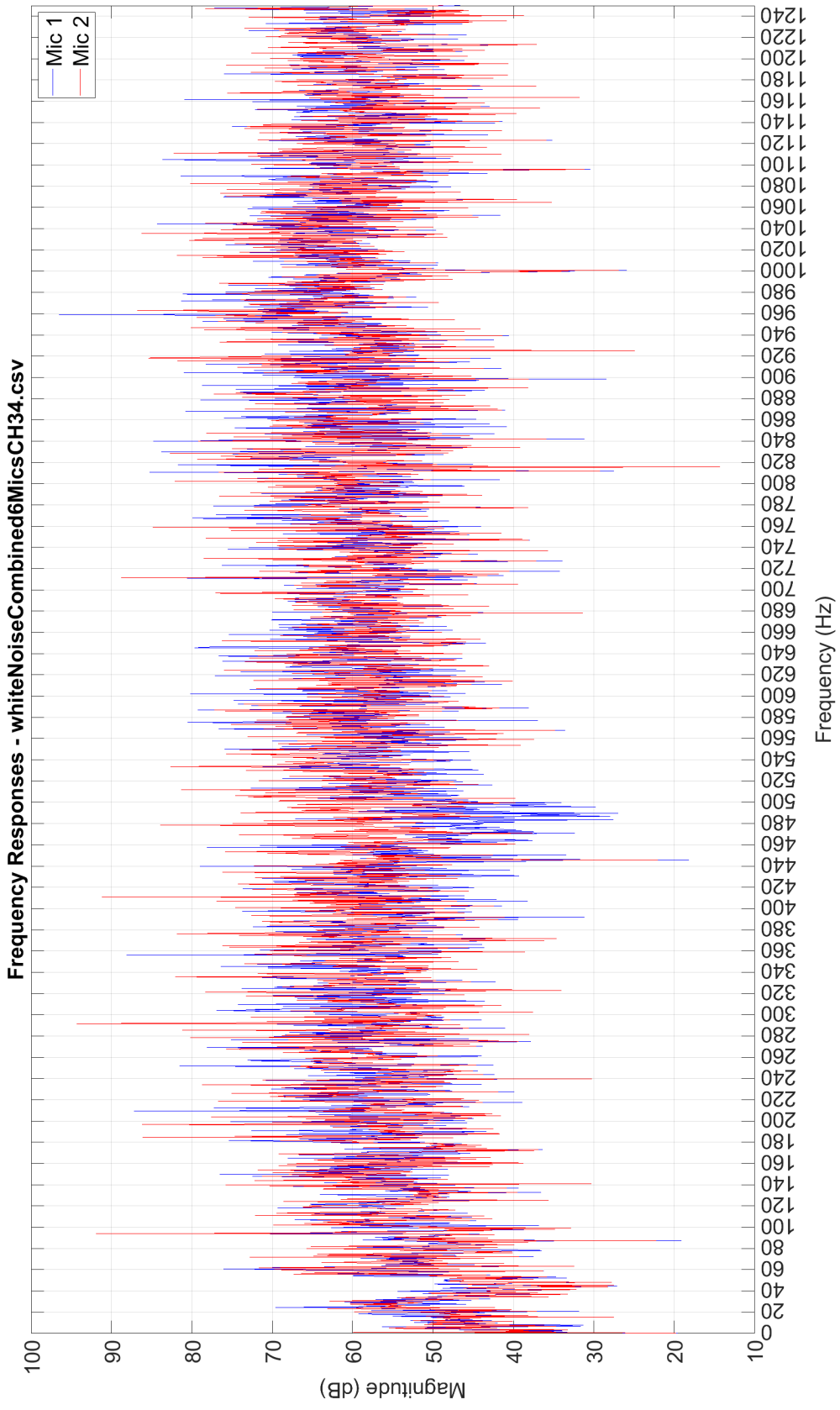


Figure M.15: Impulse response in the frequency domain of white noise on channel 3/4.

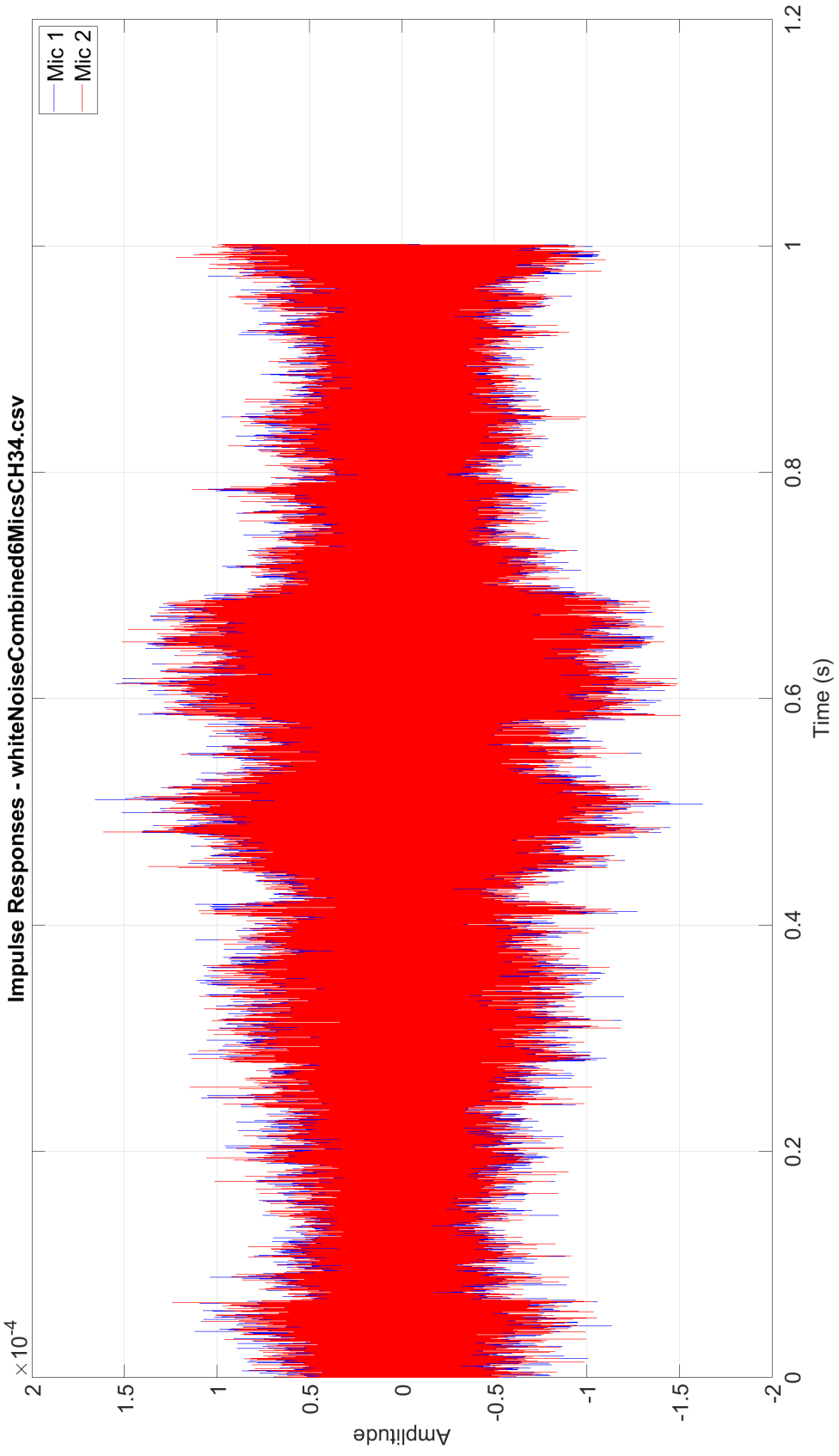


Figure M.16: Impulse response in the time domain of white noise on channel 3/4.

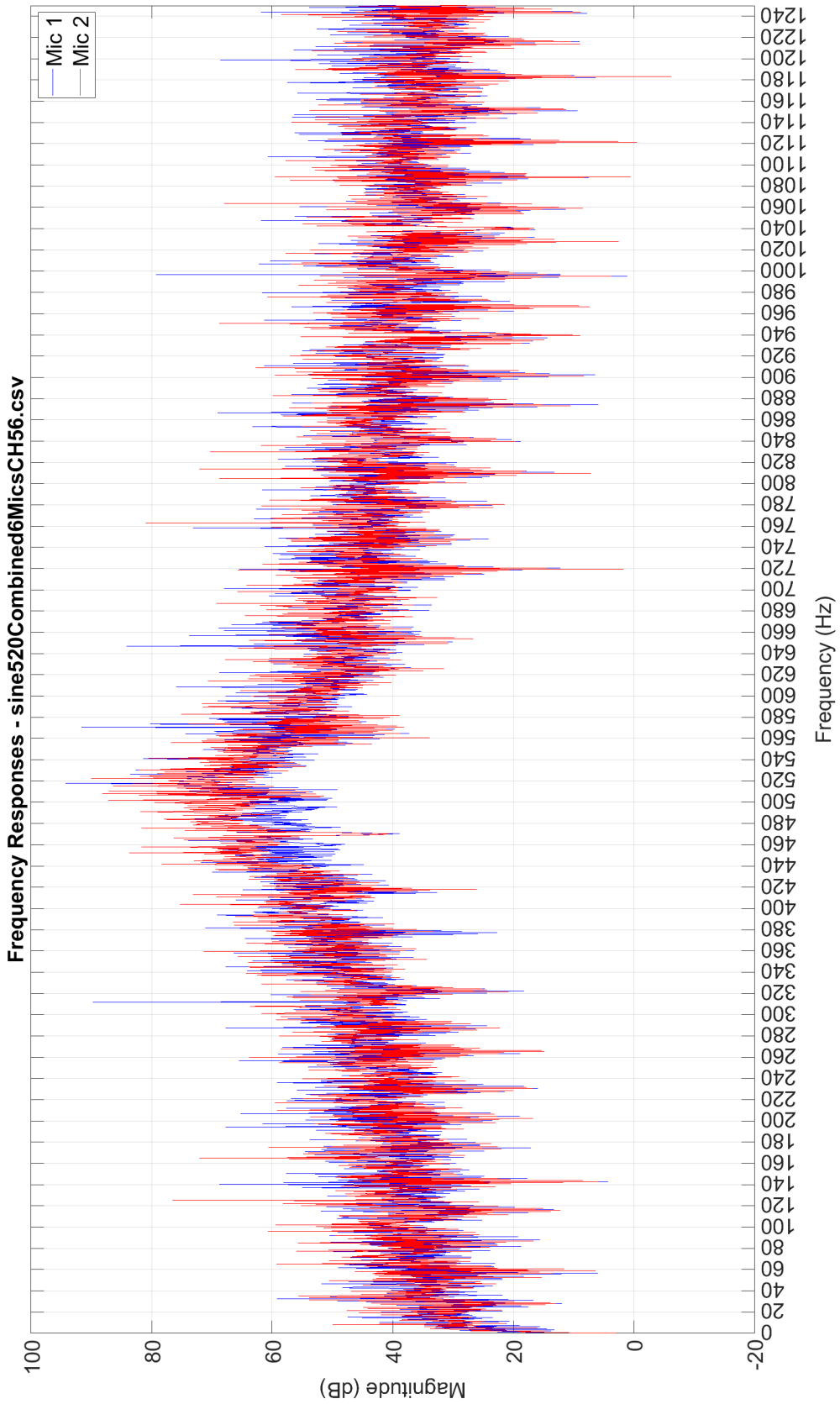


Figure M.17: Impulse response in the frequency domain of channel 5/6 at 520Hz.

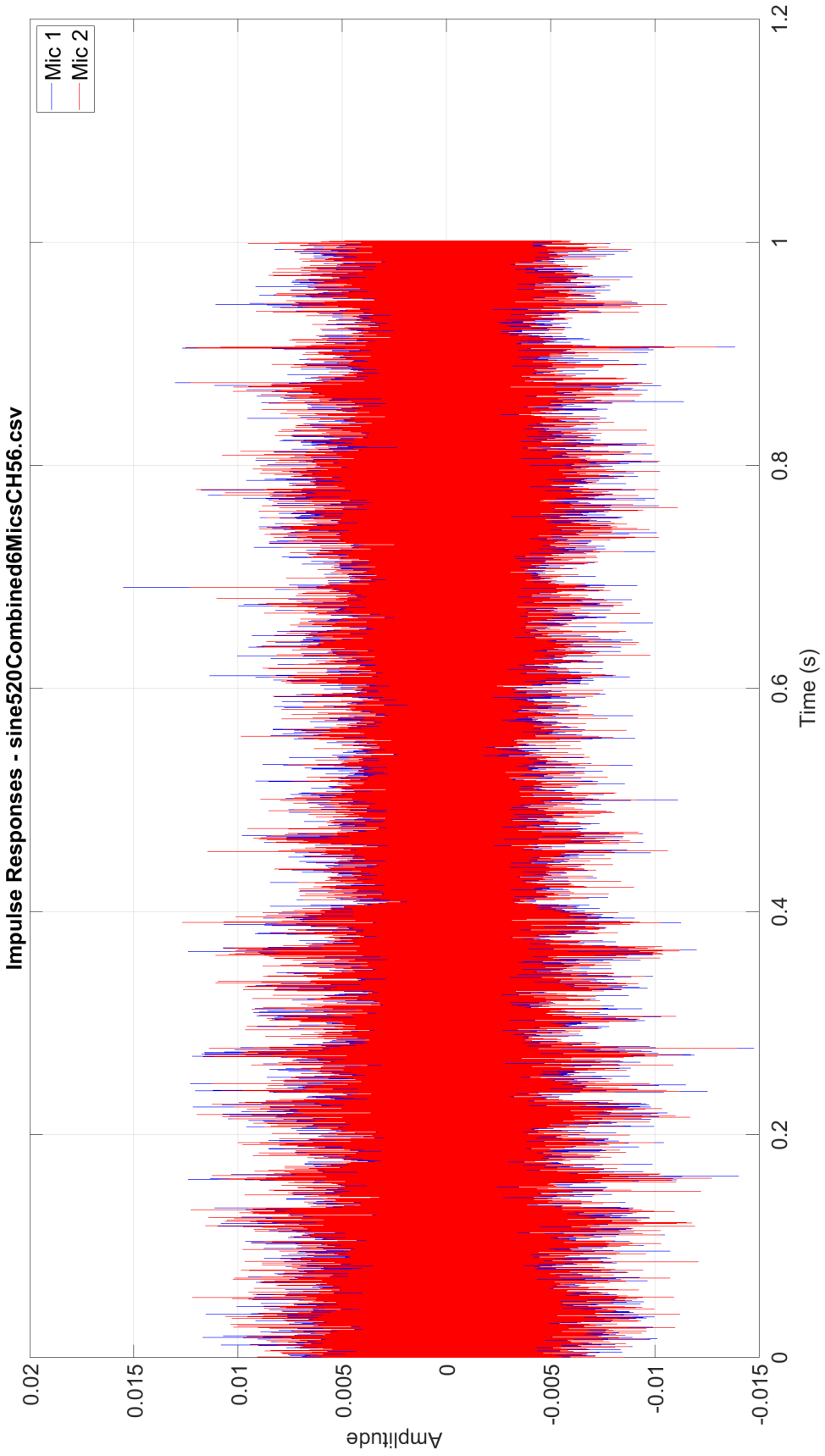


Figure M.18: Impulse response in the time domain of channel 5/6 at 520Hz.

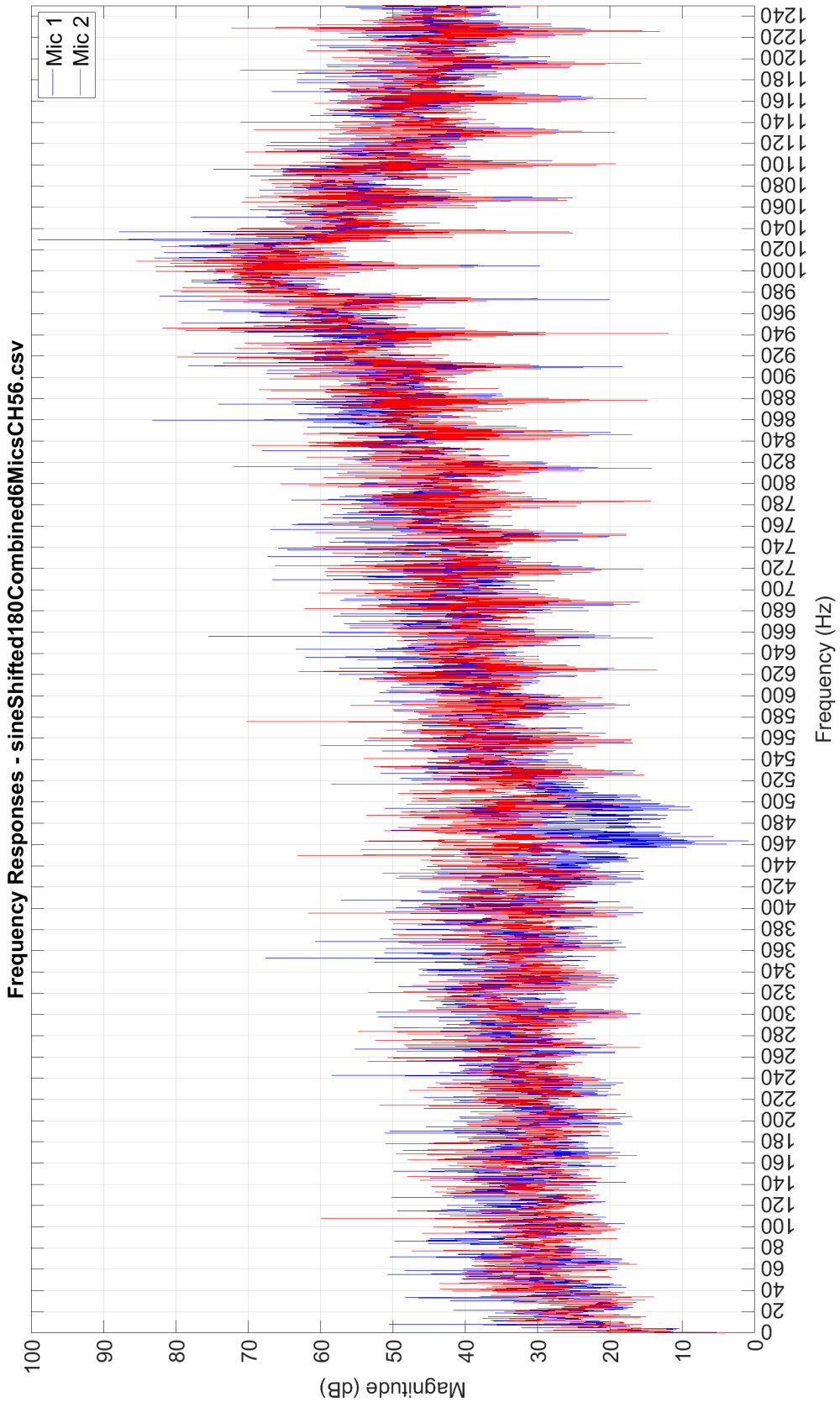


Figure M.19: Impulse response in the frequency domain of channel 5/6 at 1kHz phase shifted 180 degrees.

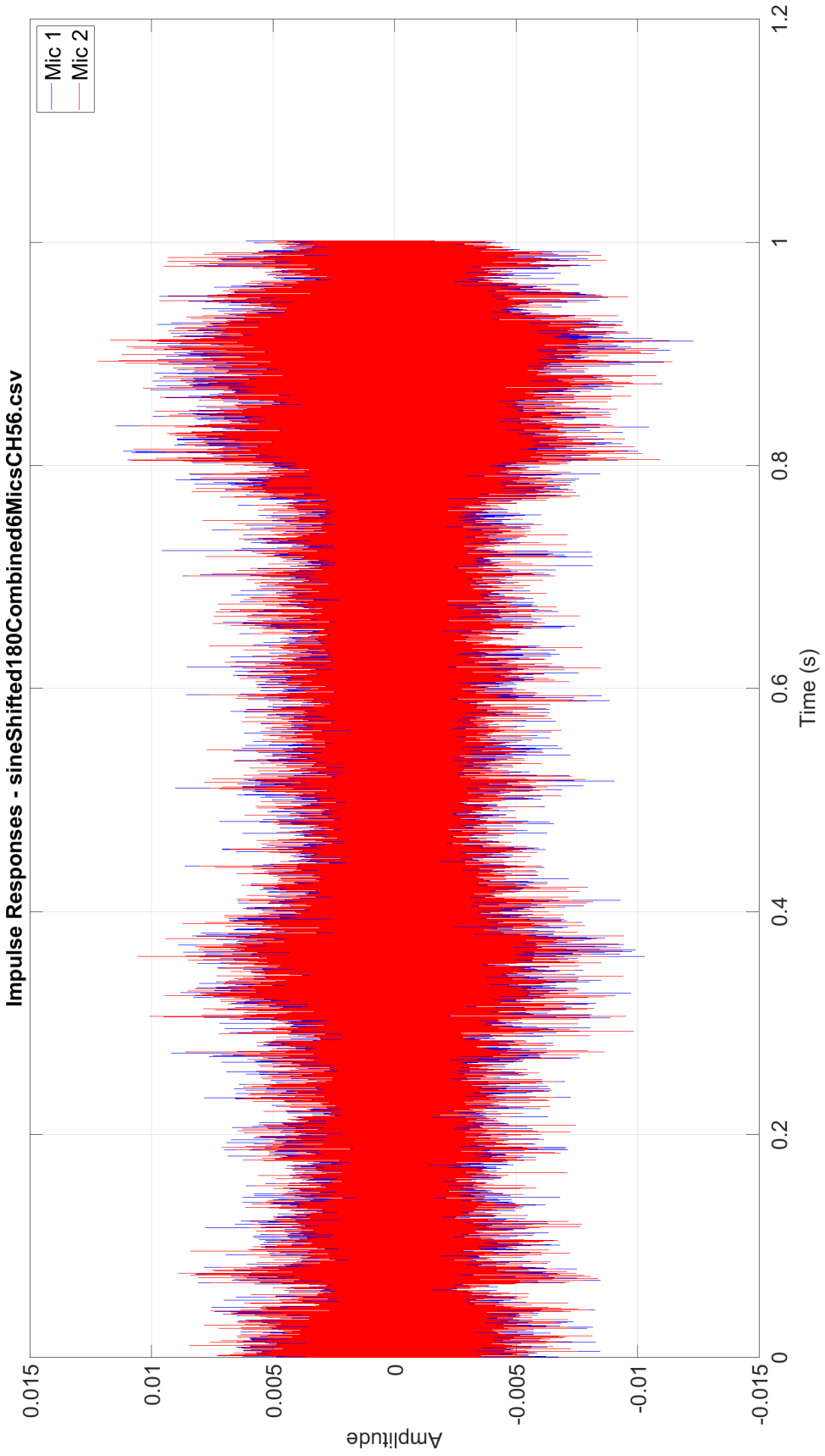


Figure M.20: Impulse response in the time domain of channel 5/6 at 1kHz phase shifted 180 degrees.

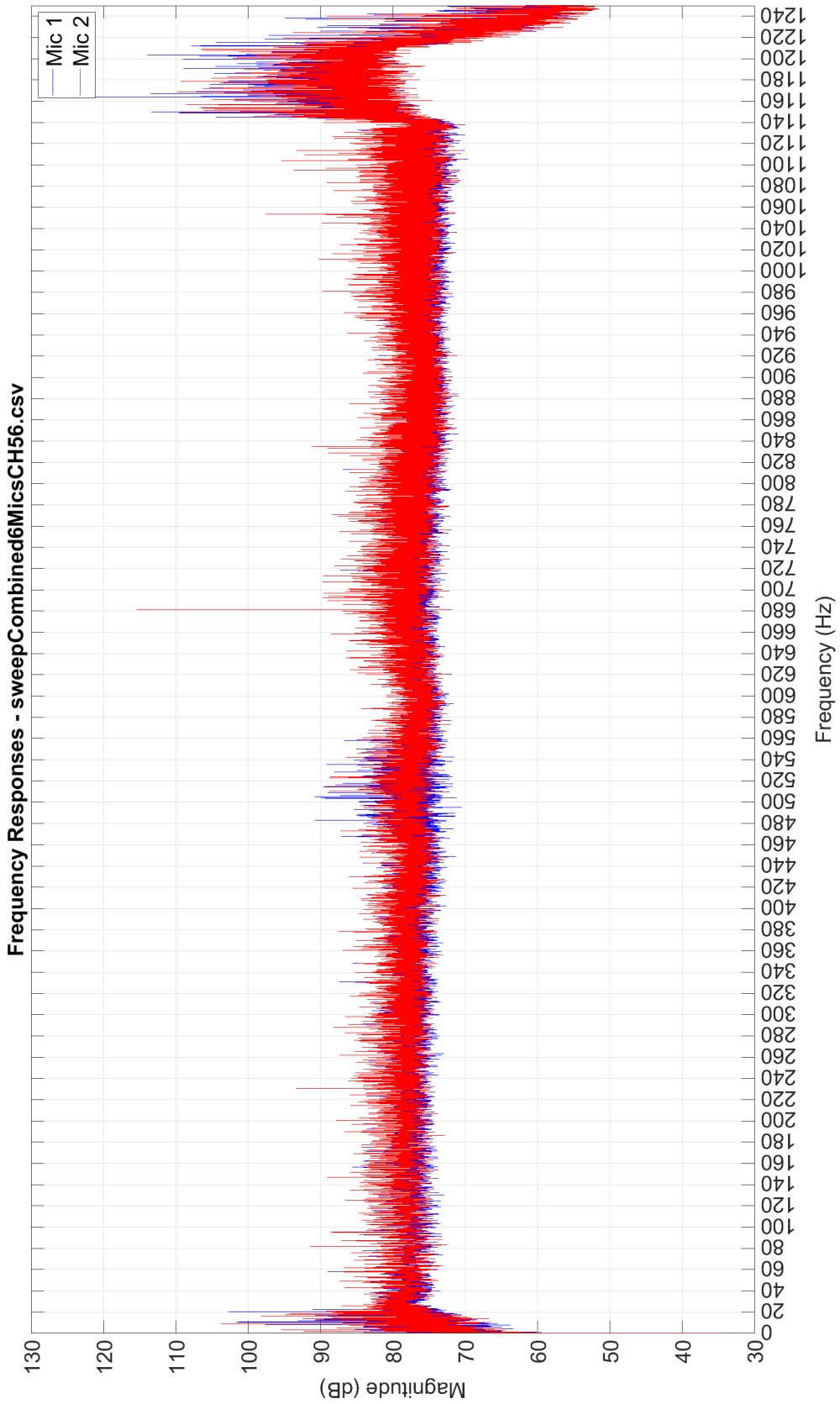


Figure M.21: Impulse response in the frequency domain of a sine sweep from 20Hz to 1220Hz on channel 5/6.

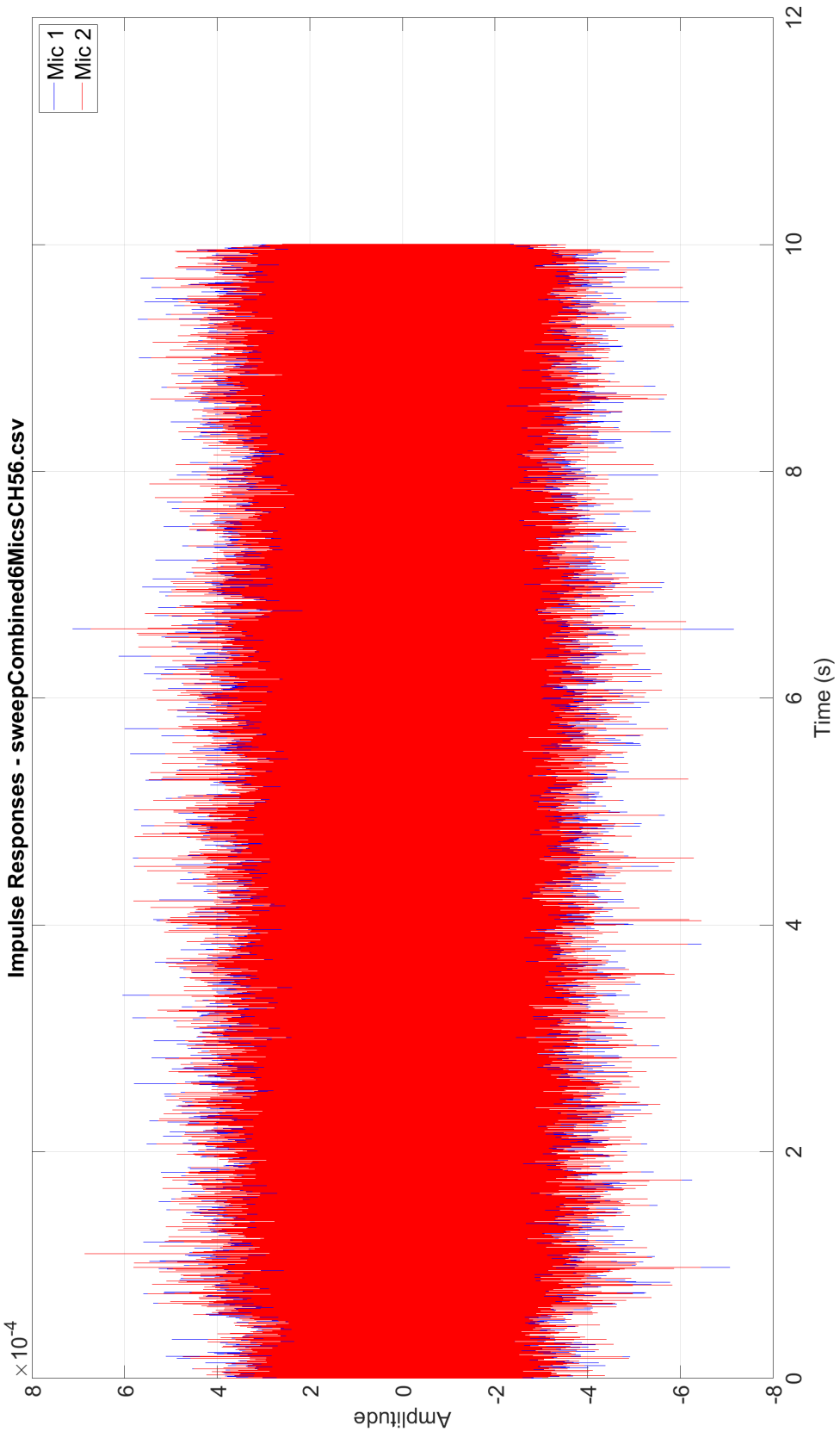


Figure M.22: Impulse response in the time domain of a sine sweep from 20Hz to 1220Hz on channel 5/6.

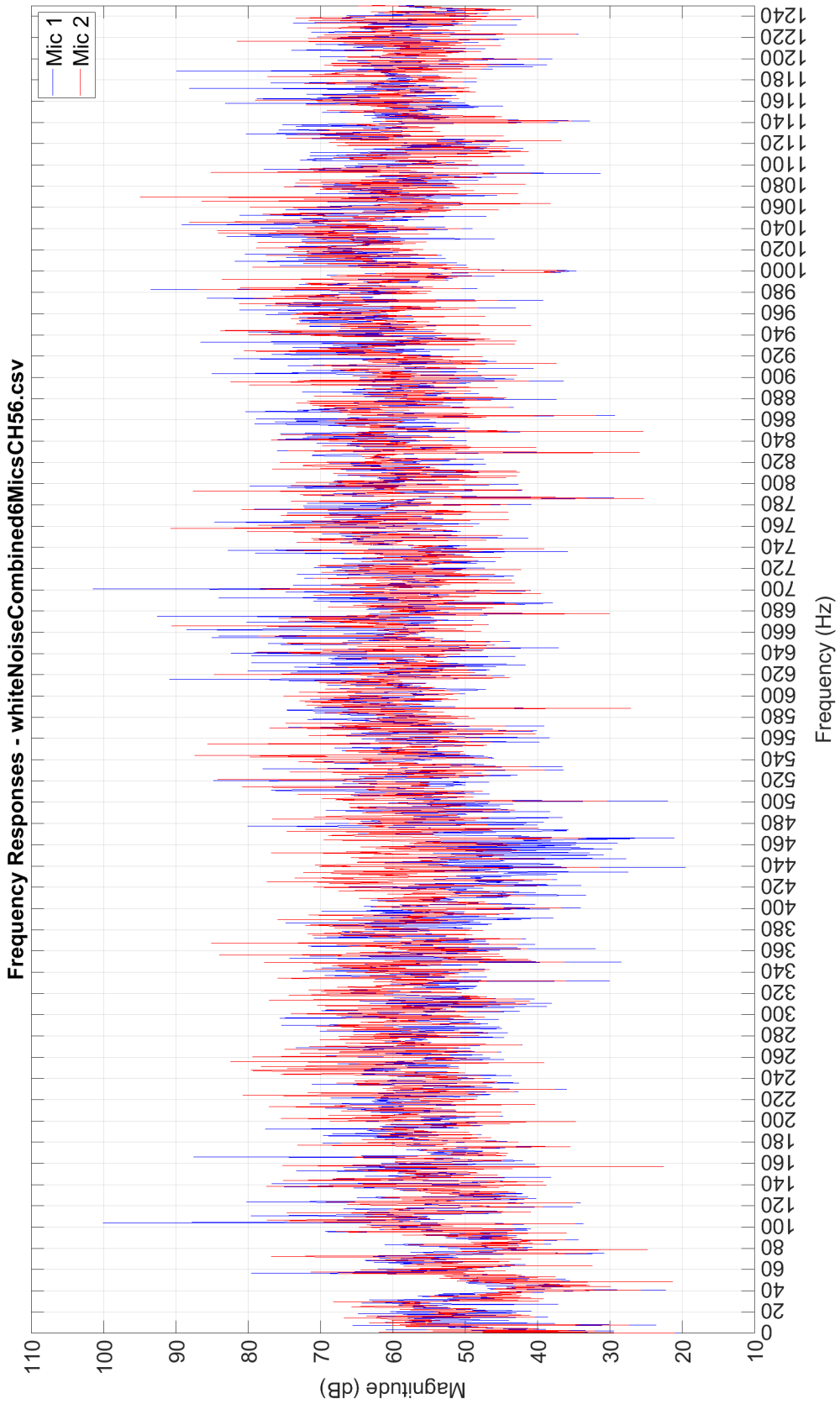


Figure M.23: Impulse response in the frequency domain of white noise on channel 5/6.

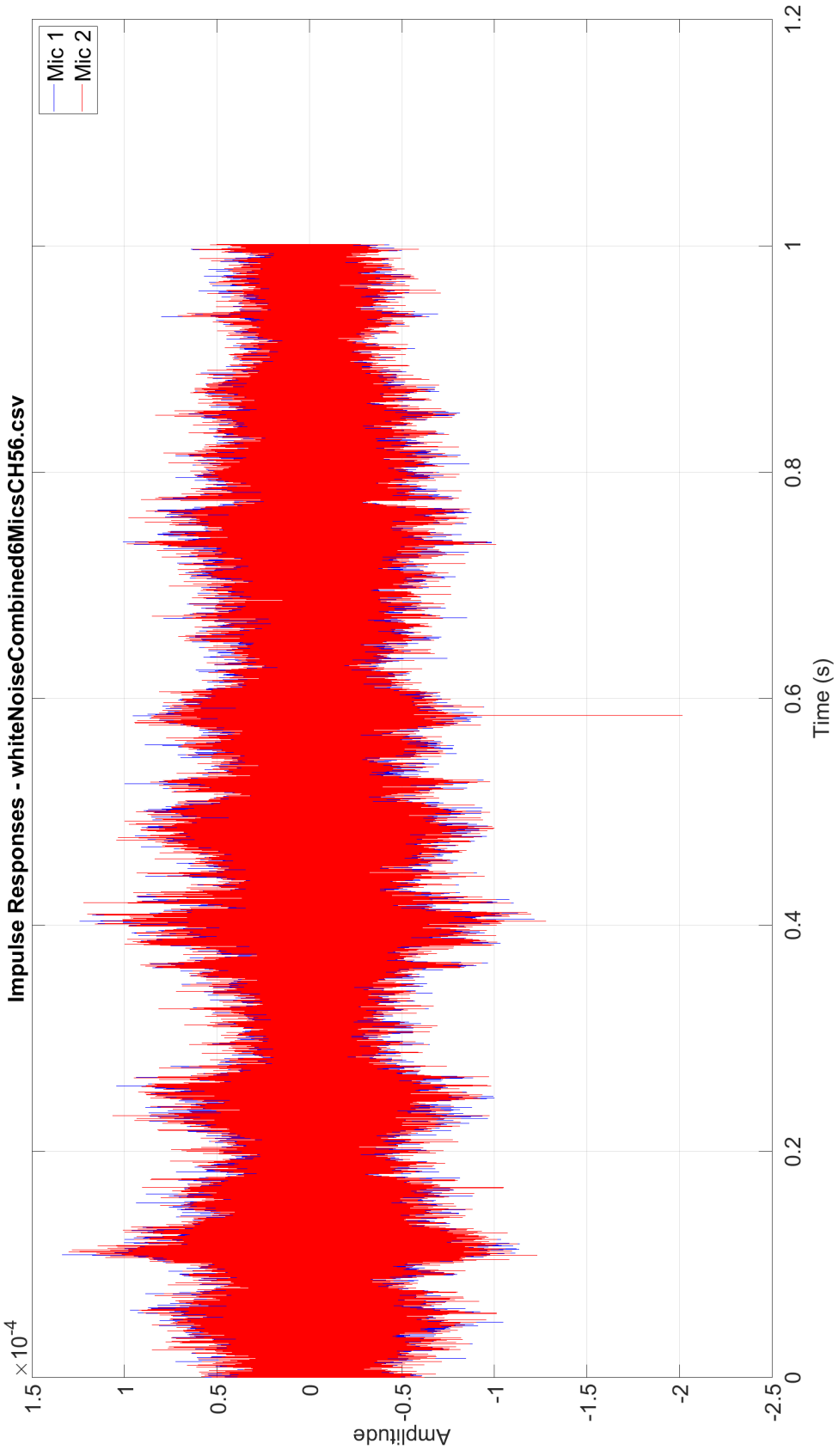


Figure M.24: Impulse response in the time domain of white noise on channel 5/6.