DESIGN AND IMPLEMENTATION OF NONDESTRUCTIVE EVALUATION INSTRUMENT FOR ACOUSTICS-BASED PREDICTION OF WOOD MODULUS OF ELASTICITY AND A METHOD TO ASSESS THE RECORDED LONGITUDINAL STRESS WAVE

UNDERGRADUATE THESIS

in partial fulfillment of the requirements
for the degree of Bachelor of Engineering
in Engineering Physics

submitted by

ABDURRACHMAN MAPPUJI
12/329912/TK/39124

to

DEPARTMENT OF NUCLEAR ENGINEERING AND ENGINEERING PHYSICS
FACULTY OF ENGINEERING
UNIVERSITAS GADJAH MADA

2017

This page is intentionally left blank

This page is intentionally left blank

This page is intentionally left blank

I dedicated this thesis

for my parents, my brother, and my big family.

Words can’t describe your endless love.

PREFACE

All the praises and thanks be to Allah because of the gift of His grace I can finish the preparation of this thesis. Blessing along with greetings may always deliver to our prophet Muhammad SAW, to his family, his friends, his ummah until the end of time, amen.

This thesis is submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Engineering Physics, Universitas Gadjah Mada. The title proposed is “Design and Implementation of Nondestructive Evaluation Instrument for Acoustics-Based Prediction of Wood Modulus of Elasticity and A Method to Assess Its Recorded Longitudinal Stress Wave”.

This work would not have been possible without the advice and support of many people. First, and foremost, I would like to thank my advisor *Bapak* Balza Achmad, ST., M.Sc.E. and *Ibu* Sentagi Sesotya Utami, ST., M.Sc., Ph.D. for providing me the opportunity and guidance throughout this project. Great thanks go to Dr. Ali Awaludin for his immense support and providing space in his laboratory for setting up some of my experiments. Special thanks go to *Ibu* Faridah ST., M.Sc., and *Bapak* Dr. Gea Oswah Fatah Parikesit, for serving on my thesis committee and providing advice and review for the thesis. Great thanks to Ahmad Sony Alfathany, Muhammad Fahreza, and Ridlo Qomarullah for your moral support throughout finishing my final project. I would also like to thank the Department of Nuclear Engineering and Engineering Physics Universitas Gadjah Mada for giving me the opportunity to study in their undergraduate program.

Yogyakarta, 11 April 2017,
Author

Abdurrachman Mappuji

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

E	[MPa]	=	Modulus of elasticity
MOE	[MPa]	=	Modulus of elasticity
MOE_s	[MPa]	=	Static modulus of elasticity (MOE obtained from the static test)
MOE_d	[MPa]	=	Dynamic modulus of elasticity (MOE obtained from the dynamic test is conventionally called the dynamic MOE)
MOR	[MPa]	=	Modulus of Rupture
f_0	[kHz]	=	Fundamental natural frequency of wood
L	[m] or [in]	=	Length of a wood beam
ρ	[kg/m^3]	=	Density of a material
σ	[MPa]	=	Stress applied to a material
ε	[1]	=	Strain applied to a material
P	[lb.]	=	Applied load in static flexure test setup
I	[in^4]	=	Moment of inertia in static flexure test setup
δ	[in]	=	Midspan deflection in static flexure center point loading test setup
C	[m/s]	=	Acoustic velocity
C_T	[m/s]	=	Acoustic velocity of wood measured with ultrasonic method
C_L	[m/s]	=	Acoustic velocity of wood measured with resonance longitudinal method
h	[kHz]	=	Damping factor
ζ	[1]	=	Damping ratio
ω_n	[kHz]	=	natural frequency
R^2_{ed}	[1]	=	The coefficient of determination of the MOE_s vs. MOE_d
R^2_{envl}	[1]	=	The coefficient of determination of the exponential decay envelope fitting procedure

INTISARI

Modulus elastisitas (MOE) kayu merupakan nilai yang dapat memprediksi karakteristik mekanis kayu. Nilai ini dapat memprediksi dampak retakan kayu dan kerusakan internal terhadap kekuatan keseluruhan. Namun, masih jarang ditemukan kayu terklasifikasi tinggi di Indonesia karena kompleksitas evaluasi menggunakan metode konvensional berdasarkan evaluasi merusak (DT) di mana sampel yang diuji sengaja dihancurkan atau evaluasi non destruktif (NDE) statis di mana alat berat dibutuhkan. Penelitian ini menggunakan metode NDE untuk mendapatkan nilai MOE dan mengusulkan instrumen genggam untuk melakukan uji dinamis berdasarkan metode gelombang tegangan longitudinal (LSWM), metode populer di antara beberapa NDE untuk menguji nilai MOE. Penelitian sebelumnya menunjukkan bahwa untuk berbagai kayu lokal yaitu Sonokeling, Jati, Sukun, Akasia, Munggur, dan Mahoni LSWM memiliki kinerja yang baik. Metode ini memiliki korelasi rata-rata MOEd dinamik terhadap nilai MOE yang diperoleh dari metode statis (MOE_s) atau R^2 sebesar 0.898. Dalam penelitian ini instrumen dikembangkan berdasarkan versi LSWM yang disempurnakan. Dalam versi yang disempurnakan ini, selain memanfaatkan frekuensi natural dasar, f0, sebagai parameter non-destruktif, diusulkan parameter untuk menilai seberapa besar gelombang yang tercatat mewakili gelombang tegangan longitudinal aktual. Dalam metode ini, pengukuran gelombang tercatat dilakukan dengan melakukan filter band-pass dengan frekuensi tengah pada f0 dan mengekstrak sampul respon impuls dari gelombang tegangan longitudinal yang telah difilter dan melakukan *fitting* eksponensial pada sampul yang diekstrak. Instrumen tersebut menggunakan komputer mikro Raspberry Pi 2 sebagai prosesor sinyal, layar sentuh LCD, soundcard USB dan mikrofon dinamis dengan rentang frekuensi 0.1-5 kHz dan -64 Â± 3dB. Instrumen ini menyediakan perhitungan NDE semi-otomatis dan instrumen genggam berdimensi kecil 12x8x4 cm dan didukung dengan catu daya USB 1200 mAh. Desain secara keseluruhan ringan, praktis, mudah digunakan, dengan penyimpanan internal untuk kemampuan transfer data dan yang paling menguntungkan adalah kemampuan evaluasi tak merusak untuk memprediksi modulus elastisitas kayu. Sebuah parameter baru bernama R^2_{envl} diusulkan. Ini adalah koefisien determinasi dalam perhitungan rasio redaman. Setelah sinyal yang direkam diperoleh untuk pengukuran MOE_d, kami menghitung rasio redaman. Jika R^2_{envl} rendah diperoleh, dapat disimpulkan bahwa ada probabilitas tinggi bahwa data yang direkam tidak dapat mewakili sinyal gelombang tegangan longitudinal dan pengukuran yang perlu dilakukan kembali.

Kata kunci – nondestructive evaluation, longitudinal stress wave, instrumentation, wood

ABSTRACT

Modulus of elasticity (MOE) of a wood is a value that can predict the mechanical characteristics of a wood. It can predict the impact of a wood crack and internal damage to the overall strength. However, it is still rare to find a high graded wood in Indonesia due to the complexity of the evaluation using a conventional method based on Destructive Evaluation (DT) where the sample tested are purposely destroyed or static Nondestructive Evaluation (NDE) where heavy tools are required. This research examined an NDE method to obtain MOE values and proposed a handheld instrument to conduct dynamic test based on longitudinal stress wave method (LSWM), a popular method among several NDE to examine the MOE values. Earlier researchers show that for various local wood i.e. Sonokeling, Teak, Sukun, Acacia, Munggur, and Mahogany LSWM has a good performance. It has an average correlation of dynamic MOE (MOE_d) to the MOE values obtained from the static method (MOE_s) or R-squared of 0.898. In this research an instrument was developed based on a proposed enhanced version of the LSWM. In this enhanced version, aside from utilizing the fundamental natural frequency, f_0, as the non-destructive parameter, a method, and parameter to assess how much the recorded wave representing the actual longitudinal stress wave is introduced. In this method, the recorded wave assessment is done by performing band-pass filter with center frequency on f_0 and extracting the envelope of an impulse response of the filtered longitudinal stress wave and perform fitting to the extracted envelope. The instrument uses a Raspberry Pi 2 microcomputer as the signal processor, an LCD touchscreen, a USB soundcard and a dynamic microphone with a frequency range of 0.1-5 kHz and -64 ± 3dB. It provides semi-automatic NDE calculation and a handheld mobile instrument given the small dimension of 12x8x4 cm and powered with a 1200 mAh USB power supply. The design in overall is less in weight, practical, easy to use, with an internal storage for data transfer capability and the most benefit is the NDE capability to predict modulus of elasticity of woods. A new parameter named R^2_{envl}. It is the coefficient to determine the envelope fitting in the damping ratio calculation. After the recorded signal is obtained for the MOE_d measurement, we calculate the damping ratio. If a low R^2_{envl} is obtained, it is concluded that there is a high probability for the data of not being able to represent the longitudinal stress wave signal and measurement needed to be redone.

Keywords – nondestructive evaluation, longitudinal stress wave, instrumentation, wood

CHAPTER I
INTRODUCTION

I.1 Background

Indonesia is one of the top wood exporters. Large forest cover is one of the factors. It has 46.33 % forest cover or 88.17 million ha of forest [1]. According to the data from World's Richest Countries [2] in 2015, Indonesia exported 3.2 % of the world's wood exports or equivalent to $4 billion and ranked 7th among the top wood exporters. This number is increasing from 2014 which only exported 2.9 % of the world's wood exports [3]. It is a strong signal that wood is very potential in Indonesia.

Wood is a versatile material; its product varies from structure to art. The quality of the wood is important when choosing a wood material. We can assess the quality of wood both qualitatively and quantitatively. The qualitative method provides quick assessment, but less accurate. It only involves some inspections usually visual inspections; even in some cases in Indonesian people only determine the quality of wood by its name. The qualitative method cannot accurately determine the quality of wood because besides vary between trees, the properties of wood also vary in the same tree.

The quantitative method provides a more accurate assessment. It involves some rigorous procedures, sometimes complex and causes the assessment slower in some cases. Indonesia National Standard number 7973-2013 was published and regulates a grading system standard for assessing the wood quality [4]. One of the key parameters is the modulus of elasticity (MOE). The MOE is the stiffness of a material. It is how easily it is bent or stretched [5]. Based on the standard, we can grade the quality of wood by its MOE.

There are two methods for evaluating the MOE i.e. destructive evaluation and nondestructive evaluation. The destructive evaluation involves some samples to be destructed as a representation of the whole population member. It also uses some complex measuring instruments which lead to a slow evaluation process. In most cases, nondestructive evaluation provides quicker evaluation process as well as cost-efficient evaluation. It doesn't need to destroy the measurement sample to evaluate the wood.

Summarizing from [6], for wood, there are some nondestructive evaluation methods, i.e. static bending, transverse vibration, and longitudinal stress wave. Static bending is the most widely adopted method for determining the modulus elasticity of wood. In this method, we measure modulus of elasticity by giving a specific load to a simply supported wood and measure how much the load bent the wood. The main disadvantage of this method is heavy tools are required to conduct the evaluation.

Transverse vibration is a dynamic evaluation method where an electrically controlled motor induces a transverse vibration in wood and an LVDT sensor measure the vibration and an oscilloscope is used to see the vibration characteristics. The main disadvantages of this method are several measuring instruments to conduct the evaluation is required.

Longitudinal stress wave method is a simple method where we initiated a longitudinal stress wave with an impulse input to a simply supported wood in one end and record the signal in near field area of another end with a microphone. In Indonesia alone, this method is well researched. Feliana in 2014 researched the possibility to examine the correlation between modulus of elasticity measured with this method to modulus of rupture [7]. Ayutyastuti in 2015 researched the performance of this method to measure the modulus of elasticity in various local wood in Indonesia [8]. The main advantages of this method are the tools required to conduct this evaluation is easy to find. It only needs an impactor, computer, soundcard, and microphone where the last three tools are can be packaged in one device. However, since we record the signal acoustically with a microphone, some additional noises may involve as we will discuss further in the discussion.

In the previous works mentioned, the MOE is predicted using generic empirical formula which involves the fundamental natural frequency of wood as shown in (1).

$$MOE_d = 4 f_0^2 L^2 \rho \tag{1}$$

where MOE_d is the dynamic modulus of elasticity (MOE obtained from the dynamic test is conventionally called the dynamic MOE), f_0 is the fundamental natural frequency of wood, L is the length of the wood and ρ is the density of wood. The fundamental natural frequency acquired by tapping one end of simply supported wood and performing the fast Fourier transform to the recorded signal from the microphone on the other end.

The main part of this evaluation is measuring the fundamental natural frequency. Commonly the measurement done with several tools mentioned earlier – a microphone, a soundcard, and a computer. A microphone received the acoustic signal and convert it to an electric signal, the soundcard sampled the signal and pass the signal to computer, and computer transform the signal to frequency domain to determine the f_0. Technically, it is possible to combine those three tools to a single instrument and make it possible to measure the f_0 faster and make the instrument small enough to carry. We can also embed a generic LSWM formula to it. This kind instrument may accelerate the adoption of this NDE method and would empower wood industry to provide a high-quality wood product to its consumer.

However, an attention must be paid to how we initiate a longitudinal stress wave. The tapping characteristic is expected to be close to a unit impulse, thus, the resulting longitudinal stress wave is its actual natural response. The wood beams are said to be simply supported if their supports create only the translational constraints. In simply supported setup, ideally, the supports should be rigid so that no vertical displacement of the supports occurs. However, Some times the translational movement may be allowed in one direction with the help of rollers [6].

While using, the instrument described, the user is expected to set up the measurement by themselves, and some mistakes often occur when setting up the experiment e.g. the tapping characteristics is far from unit impulse, the support is not rigid or measuring in a noisy environment. These mistakes lead to a problem of recorded signal is not representing the longitudinal stress wave. A parameter to measure how well the recorded signal representing the longitudinal stress wave of wood is required. Hence, we will be able to justify how much we confident with the prediction and provide a more accurate measurement.

I.2 Problem Statement

The problem statement can be formulated as follows,

(i) A nondestructive evaluation instrument which measure the fundamental natural frequency of wood which embedded with generic LSWM formula for predicting the wood modulus of elasticity is needed to be designed and implemented.

(ii) A parameter to measure how well the recorded signal representing the longitudinal stress wave of wood is need to discover to increase the accuracy of the measurement of fundamental natural frequency.

I.3 Objectives

The objectives of this research are,

(i) To build a prototype of the instrument which measure the fundamental natural frequency of longitudinal stress wave of wood.

(ii) To provide a method to measure how much recorded longitudinal stress wave representing the actual longitudinal stress wave.

(iii) Include the method in objective (ii) to the instrument prototype.

I.4 Benefits

The result of this research is expected to be able to give benefits as follows,

(i) Provide a way to predict MOE by designing and implementing a simple and practical instrument to measure the fundamental natural frequency of longitudinal stress wave of wood.

(ii) Empower the wood practitioners and the society to deliver and get a high-quality wood and its product.

(iii) Educate data literacy, especially in the wood industry, because today's people in Indonesia buy unclassified wood, they simply buy wood by its name or visual inspection.

CHAPTER II
LITERATURE REVIEW

II.1 The Modulus Elasticity of Wood

II.1.a Introduction

The MOE of a material is a fundamental property of every material that invariable. It is dependent upon temperature and pressure, however. The MOE is the stiffness of a material. In other words, it is how easily it is bent or stretched [5]. The mathematical equation for the MOE can be written as in (2).

$$MOE = \sigma / \varepsilon \quad (2)$$

where,

$$\sigma = stress = \frac{force}{cross\_sectional\_area} = F / A$$

$$\varepsilon = strain = \frac{change\_in\_length}{original\_length} = \Delta L / L_0$$

When graphed, the resulting plot is similar the Figure 1. The MOE is the slope of the initial section of the curve, which forms a linear line, $y = m \cdot x + b$.

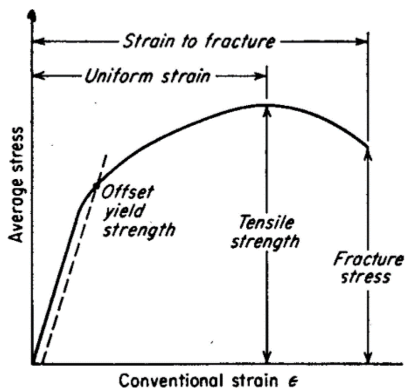


Figure 1. The correlation between stress and strain in materials is varying in different strain level [5].

When a material reached a certain stress, the material will begin to deform. It is up to point where the structure of the material is stretching and not deforming. However, if we are to stress the material more than this, the molecules or atoms inside will begin to deform and permanently change the material.

A good analogy to this would be a rubber band: when we stretch a rubber band we are not deforming it, but stretching it. However, if we pull it too hard the rubber band will begin to deteriorate or deform. Usually, when this happens, it is not too much longer until it breaks.

II.1.b Static Flexure Method for Measuring the MOE

Based on how the MOE of a material evaluated, especially for wood, there are two kinds of MOE, i.e. static MOE or MOE_s and dynamic MOE or MOE_d. The MOE_s is measured via static evaluation, i.e. static flexure (often called static bending) method and the MOE_d is measured via dynamic evaluation, i.e. resonant flexure method, resonant LSWM.

The most common method for evaluating static MOE is static flexure test or also known by a static bending test. Summarizing from [6] measuring the MOE of the member by static bending methods is a relatively simple procedure that involves the load-deflection relationship on a simply supported beam. Modulus of elasticity can be computed directly by using equations derived from fundamental mechanics of materials. This method is widely implemented in some commercially available equipment. There are two commonly used flexure test, i.e. center point loading setup and alternative two-points bending setup.

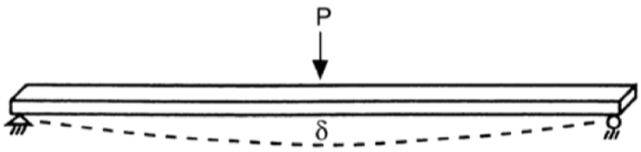


Figure 2. Flexure center point loading test setup [6].

Figure 2 illustrates a typical flexure center point loading test setup. The specimen is simply supported at its ends, a load applied, and the midspan deflection that results from the load measured. MOE of the specimen is calculated using (3)

$$MOE = {PL^3}/{48I\delta} \quad (3)$$

where P is applied load (lb.), L is span (in), I is moment of inertia (in^4), and δ is midspan deflection (in).

The Figure 3 shows an alternative two points bending setup. The only difference between this setup is that equal loads are applied at a known distance from each end. MOE of a specimen is calculated using (4).

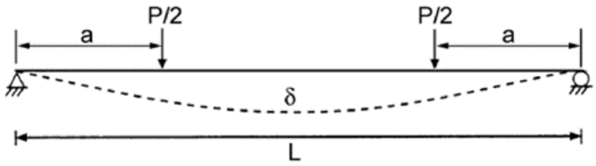


Figure 3. Alternative bending test setup [6].

For both test setups, significant attention must be placed on the design of the end supports. Ideally, the supports should be rigid so that no vertical displacement of the supports occurs. In addition, horizontal movement of the specimen on the end support should not be restricted, and the end supports need to be rotated and fixed to accommodate twist in the specimen.

$$MOE = \frac{Pa(3L^2 - 4a^2)}{48I\delta} \tag{4}$$

II.1.c The Problem with Static Flexure Method

Suppose we want to measure the MOE of wood during a visit to a wood shop, wood laboratory, or similar place with limited equipment. The static flexure method for measuring the MOE at least needs two things, viz, a load generator (e.g. an iron block, electrically induced load), and an LVDT position sensor (for measuring the δ). This method is inefficient in term of the measurement procedures.

II.2 The Longitudinal Stress Wave Method (LSWM)

We now give a brief introduction to longitudinal stress wave method. The main purposes of this section are to introduce notation and concepts in a simple and (hopefully) familiar context nondestructive evaluation of wood.

II.2.a Introduction

The longitudinal stress wave method for measuring the MOE is the nondestructive method which utilizes the acoustical properties of wood. It utilizes the acoustic velocity of induced stress wave in a material as NDE parameter [6]. The MOE can be computed using the acoustic velocity *C*, and the mass density of the wood beam ρ as shown in (5).

$$MOE_d = C^2 \rho \tag{5}$$

There are several ways to compute *C*. For wood beam and logs, we can use the longitudinal wave velocity. Ultrasonic approach and resonance-based approach are the most popular technique to compute *C*. Basically, in the ultrasonic approach we are counting time of the induced stress wave to propagate from one end to another end. Longitudinal stress wave induced by the ultrasonic transmitter on the one end and received by the ultrasonic receiver on the other end. The timer counts the travel time. The longitudinal wave velocity computed using (6).

$$C_T = \frac{s}{\Delta T} \tag{6}$$

where C_T is the acoustic velocity (m/s), *s* is the distance between the two ultrasonic probes (m), and ΔT is the time of flight counted by the timer.

The resonance-based approach is the most widely adopted approach. One of the main contributors of this approach is Japanese researcher Sobue in 1986 [9]. In this approach, a stress wave is initiated by a mechanical impact (e.g. from a hammer) on the one end, and the longitudinal stress wave are subsequently recorded by a digital device. This approach is based on the longitudinal acoustic pulse acoustic pulses in wood and provides a weighted average acoustic velocity. Most resonance-based acoustic tools have a built-in Fast Fourier Transform (FFT) program that can analyze and output the natural frequencies of the acoustic signals. The acoustic velocity is then determined from (7).

$$C_L = 2 f_0 L \tag{7}$$

where C_L is the acoustic velocity of wood (m/s), f_0 is the fundamental natural frequency of an acoustic wave signal (Hz), and L is the length of the sample (end-to-end, m).

The resonance-based acoustic method is a well-established non-destructive evaluation (NDE) technique for measuring long, slender wood members such as logs, poles, and timber [6]. The inherent accuracy and robustness of this method provide a significant advantage over ultrasonic approach (time-of-flight-based) measurement in this application. In contrast to the ultrasonic approach, the resonance method stimulates many, possibly hundreds, of acoustic pulse reverberations in a wood, resulting in a very accurate repeatable velocity measurement. A complete MOE (Pa) equation for the resonance-based acoustic method can be formulated by substituting the (7) to the (5) as shown in (1). The accuracy and robustness of this method are one of the considerations to use the resonance-based approach in this research.

II.2.b Recent Developments

In Indonesia, itself, there are several researchers conducted the developments of resonance-based longitudinal stress wave method. Research by Feliana in 2014 [7] and Ayutyastuti in 2015 [8] are several latest developments in this area.

Feliana and Ayutyastuti both conducted research at the same laboratory. They explore the possibilities of LSWM to measure the modulus of elasticity and modulus of rupture of wood. Feliana studied the relationship between the MOE obtained from the static (destructive) method, MOE_s, and the modulus of rupture, MOR, with the MOE obtained from (dynamic) LSWM, MOE_d. The study uses wood from Sonokeling, Teak, Sukun, Acacia, Munggur, and Mahogany with 10 samples for each species. All the samples are 60×120×1200 mm.

Feliana observed a relationship between MOEs and MOR in a unit of MPa as shown in (8) with $R^2 = 0.74$.

$$MOR = 0.007 MOE_s - 2.5411 \tag{8}$$

Whereas the relationship of MOE_d and MOR is shown in (9) with $R^2 = 0.55$.

$$MOR = 0.0041 MOE_d + 6.0835 \tag{9}$$

The MOEs can be estimated using the (10) with $R^2 = 0.68$.

$$MOE_s = 0.5635 MOE_d + 1472.9 \tag{10}$$

This research explores the possibilities of estimating MOR and MOE_s with MOE_d obtained from LSWM. However, this research only exploring the LSWM for predicting another elastic property of wood.

Ayutyastuti with the same wood samples and method studied the relationship between MOEd and MOE_s for wood. Hence, we can understand which wood best suited with the LSWM. Table 1 shows the summary of her research.

Table 1. The coefficient of determination R^2 for the relationship between MOE_d and MOE_s [8].

Wood Type	Scientific Name	R^2
Sonokeling	*Dalbergia latifolia roxb.*	0.57
Teak	*Tectona grandis l.f.*	0.59
Sukun	*Artocarpus altilis*	0.63
Acacia	*Acacia mangium*	0.90
Munggur	*Pithecolobium saman benth*	0.90
Mahogany	*Swietenia spp.*	0.95

We can see from the result that the R^2 of the relationship between MOE_d and MOE_s for Acacia, Munggur, and Mahogany is greater or equal to 0.90 in contrast for Sonokeling, Teak, and Sukun the R^2 is less than 0.70. For the first group, we can conclude that the LSWM is best suited for estimating the MOE_s while for the second group we need to have more depth exploration to find the answer.

Ayutyastuti’s research used a BSWA omnidirectional microphone with flat response and has a frequency range of 177 Hz – 5680 Hz. The results from [7], [8] is the basic building blocks for the instrumentation design in this thesis. Hence, we used the near identical microphone used in [8] with specifications shown in Table 2.

Table 2. The specifications of microphone used in the instrumentation design.

Physical Properties	Specification
Audio Jack	3.5 mm
Plug Diameter	3.5 mm
Frequency Range	0.1 – 5 kHz
Sensitivity	-64 ± 3 dB
Length	16 cm
Weight	12 g

II.2.c The Problem with Resonance-based LSWM

Suppose we want to predict the MOE of the wood beam with resonance-based LSWM, but when we tap the one end of wood with an impactor (e.g. hammer), we tap the wood very hard and the friction of the wood with its support or friction with the impactor is initiated (simply supported setup is not achieved). The friction may result in additional noise and reduces the quality of recorded signal. A criterion that describes how much the recorded signal represents the longitudinal stress wave would help the prediction process to be more accurate. If we have this criterion, we can decide whether to repeat the procedure.

II.3 Attenuation of Induced Longitudinal Stress Wave

II.3.a Introduction

Summarizing from [10], [11] free vibrations are oscillations where the total energy stays the same over time. This means that the amplitude of the vibration stays the same. This is a theoretical concept because in real systems (e.g. in wood vibration) the energy is dissipated to the surroundings over time and the amplitude decays away to zero, this dissipation of energy is called damping. Friction in the oscillating system causes the amplitudes continuously decrease. This kind of motion is no longer periodic but nonetheless considered a vibratory movement. The damping can be external or internal as the friction occurs between the oscillating system and the environment or internally in the system.

External damping exists if resistant forces applied to the moving mass of the system. Unlike external damping, the internal damping occurs if there is an existence of a hysteresis loop in the loading – unloading diagram of the elastic element. An impactor induced a longitudinal stress wave is an example of damped free vibration with internal damping.

For a system with only internal damping, based on experimental investigations, materials can be classified into two categories: materials for which the damping depends on both amplitude and frequency, and materials for which the damping depends only on the amplitude (hysteretic damping).

The damped free vibration can be modelled with an equation of motion with mass, spring and damper element as shown in (11).

$$m\frac{d^2x}{dt^2}+c\frac{dx}{dt}+kx=0 \tag{11}$$

There are two main parameters to assess the response of damped free vibration i.e. the natural frequency, ω_n, and damping ratio, ζ. Note that we differ the symbol of natural frequency in damped free vibration (ω_n) with fundamental natural frequency in longitudinal stress wave measurement (f_0) to give emphasis that for anisotropic material, we may have some natural frequency, but the one that take in to our consideration in LSWM is the f_0. Natural frequency is the frequency of oscillation of the system without damping. It is mathematically defined as (12).

$$\omega_n=\sqrt{\frac{k}{m}} \tag{12}$$

Damping ratio is a dimensionless parameter describing how oscillation in a system decay after a disturbance. The damping ratio provides a mathematical means of expressing the level of damping in a system relative to critical damping as shown in (13).

$$\zeta=\frac{c}{c_c} \tag{13}$$

where critical damping is defined as (14),

$$c_c = 2\sqrt{km} \vee c_c = 2m\omega_n \tag{14}$$

For our case, it can be defined as the ratio of damping coefficient in the differential equation (11) to the critical damping coefficient as shown in (14). For material, the damping ratio is the acoustical properties of material. It is often used for determining the appropriate material of musical instrument.

The complete solution for equation (11) in underdamping situation is shown in (15) and complete derivation is available in Appendix 2.

$$x(t) = x_0 \cdot \exp(-ht) \cdot \cos(\omega_d t - \phi) \tag{15}$$

where h is the damping factor and $h = \zeta\omega_n$. The amplitude of the solution is a function of time and it decreases towards zero. The damped vibration is quasi-harmonic (pseudo-periodic) exponentially modulated in amplitude, and the period is called a pseudo-period. Speed and acceleration also decrease in time.

Damping ratio can determine the attenuation of damped free vibration. Calculating the damping ratio would help us determine the attenuation or exponential part of (15). There is some method to calculate the damping ratio or exponential attenuation of a damped free vibration, we will discuss the most common method in the following section.

II.3.b Logarithmic Decrement Method

There are several types of research which explore some method for determining the damping ratio of the wood samples. One most inspiring research for us is research done by Ciornei in 2009 [10]. Instead of using the LSWM for estimating the MOE, he utilizes transverse vibration method for estimating the MOE. The MOE is not directly estimated. The procedure involves the measurement of the damping ratio, and then the MOE is estimated using the measured damping ratio.

The most interesting part from [10] is the method for measuring the damping ratio. Practically, there are several methods for predicting the damping ratio of damped free vibration. Two most common methods are the logarithmic decrement and the envelope fitting. Ciornei uses the logarithmic decrement for calculating the damping ratio. The logarithmic decrement is a simple method for determining the damping ratio. The logarithmic decrement represents the rate at which the amplitude of a free damped vibration decrease. It is defined as the natural logarithm of the ratio of any two successive amplitudes [12]. The example of damped free vibration is shown in Figure 4.

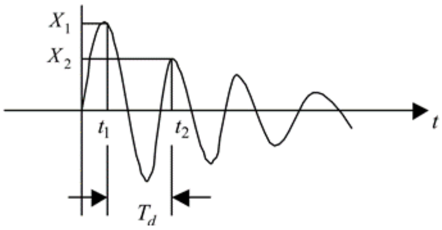


Figure 4. An example of two successive amplitudes (X_1 and X_2) of damped free vibration, we can apply the logarithmic decrement in this kind of damped free vibration.

The derivation of the logarithmic decrement is shown in (16)

$$x(t) = x_0 \cdot \exp(-ht) \cdot \cos(\omega_d t - \phi)$$

$$X_1 = A \cdot \exp(-\zeta\omega_n t_1)\,,\; X_2 = A \cdot \exp(-\zeta\omega_n t_2) = A \cdot \exp(-\zeta\omega_n (t_1 + T_d))$$

$$\frac{X_1}{X_2} = \frac{A \cdot \exp(-\zeta\omega_n t_1)}{A \cdot \exp(-\zeta\omega_n (t_1 + T_d))} = \exp(\zeta\omega_n T_d)$$

$$\delta = \ln\frac{X_1}{X_2} = \zeta\omega_n T_d \wedge \zeta = \frac{\delta}{\omega_n T_d} \tag{16}$$

For the lightly damped system, we can take X_2 or the 2nd amplitude after N cycles. The derived formula is shown in (17).

$$\frac{X_1}{X_{N+1}} = \frac{A \cdot \exp(\zeta\omega_n t_1)}{A \cdot \exp(\zeta\omega_n (t_1 + NT_d))} = \exp(\zeta\omega_n NT_d)$$

$$\delta = \ln\frac{X_1}{X_{N+1}} = N(\zeta\omega_n T_d) \vee \zeta = \frac{\delta}{N\omega_n T_d} \tag{17}$$

II.3.c The Problem with Logarithmic Decrement Method

Suppose we want to calculate the damping ratio and exponential attenuation of the longitudinal wave of anisotropic material such as wood. We can’t confidently choose two successive amplitudes from the signal because the signal contains multiple frequency components. This problem occurs when we deal with the LSWM. The signal recorded is not a pure frequency signal. Hence, we need to have another method for calculating the damping ratio damping ratio and exponential attenuation.

II.4 The SNI Wood Grading

The Indonesia National Standard (SNI) has the standard for classifying the wood by its MOE. Because the design and implementation of the instrument aim to be used particularly in Indonesia, this standard will be our reference. The complete grading classification is shown in Table 3.

Table 3. The quality grading of wood based on SNI.

Quality Code	MOE (MPa)	
	E	**E_{min}**
E25	25,000	12,500
E24	24,000	12,000
E23	23,000	11,500
E22	22,000	11,000
E21	21,000	10,500
E20	20,000	10,000
E19	19,000	9,500
E18	18,000	9,000
E17	17,000	8,500
E16	16,000	8,000
E15	15,000	7,500
E14	14,000	7,000
E13	13,000	6,500
E12	12,000	6,000
E11	11,000	5,500
E10	10,000	5,000
E9	9,000	4,500
E8	8,000	4,000
E7	7,000	3,500
E6	6,000	3,000
E5	5,000	2,500

If we want to measure an individual sample, the **E** is used for grading, but if we want to measure a population of a homogenous sample, both **E** and **E_{min}** are used instead.

CHAPTER III
THEORETICAL BACKGROUND

III.1 Waterfall Approach for System and Software Development

The development of embedded or instrumentation system requires a design process that begins with a concept and ends with the completed application [13]. While there is no such a “best way” to carry out this process, we can always choose the best-suited approach for our problem.

The waterfall approach is a series of sequential steps, as shown in Figure 5, viz, development of a specification, the creation of a preliminary design (behavioral), development into a detailed design (structural) and full implementation (physical). This approach is best-suited for this problem because the NDE method we want to implement is well established so that there is no change in requirements during the design process.

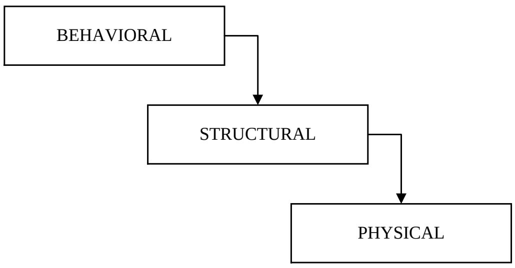


Figure 5. Waterfall design model [13]**.**

III.2 Wave Propagation in Wood

When stress or impulse induced suddenly (with an impactor) to the surface of the wood, a disturbance is generated and travels through the wood as stress wave. There are three kinds of waves are initiated by such an impact, i.e. longitudinal wave (compressive or P-wave), shear wave (S-wave), and surface wave (Rayleigh wave) (Figure 6) [6].

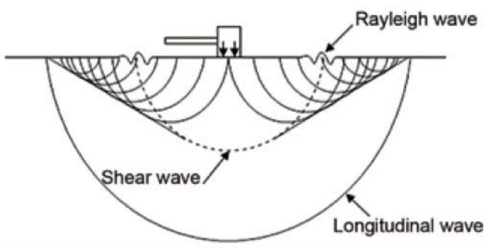


Figure 6. Three type of waves in a semi-infinite elastic material [6]**.**

A longitudinal wave initiated represents the oscillation of particles along the direction of wave propagation, such that particle velocity is parallel to wave velocity. In contrast, the motion of the particles in the shear wave is perpendicular to the direction of the propagation of the wave itself. A Rayleigh wave is usually restricted to the region adjacent to the surface. In Rayleigh wave, the particles move both up and down and back and forth, tracing elliptical paths. Although most energy resulting from an impact is carried by shear and surface waves, the longitudinal wave is the fastest wave travels and it is the easiest to detect in field applications [14]. As a consequent, the longitudinal stress wave is the most commonly used wave for material property characterization [6].

III.3 Microphone as Sensing Element

III.3.a How Microphone Works

Summarizing from [15], a microphone is a kind of transducer, a device that changes information from one form to another – change mechanical signal into electrical signal. Sound information exists as patterns of air pressure; the microphone changes this information into patterns of electric current. The recording engineer is interested in the accuracy of this transformation, a concept he thinks of as fidelity.

A variety of mechanical techniques can be used in building microphones. The two most commonly encountered in recording studios are the magneto-dynamic and the variable condenser designs.

III.3.b Dynamic Microphone

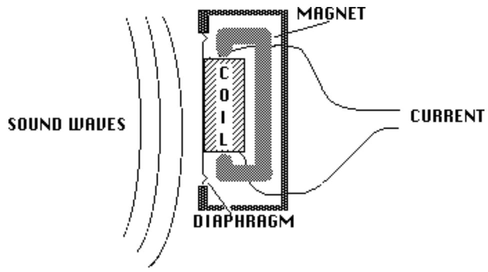


Figure 7. A dynamic microphone schematic [15]**.**

In the magneto-dynamic, commonly called dynamic, microphone, and sound waves cause movement of a thin metallic diaphragm and an attached coil of wire. A magnet produces a magnetic field which surrounds the coil, and motion of the coil within this field causes current to flow. The principles are the same as those that produce electricity at the utility company, realized in a pocket-sized scale. It is important to remember that current is produced by the motion of the diaphragm and that the amount of current is determined by the speed of that motion. This kind of microphone is known as velocity sensitive [15].

III.3.c Condenser Microphone

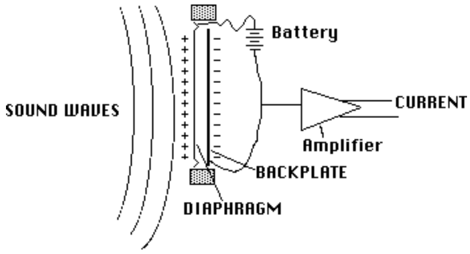


Figure 8. A dynamic microphone schematic [15].

In a condenser microphone, the diaphragm is mounted close to, but not touching, a rigid backplate. (The plate may or may not have holes in it.) A battery is connected to both pieces of metal, which produces an electrical potential, or charge, between them. The amount of charge is determined by the voltage of the battery, the area of the diaphragm and backplate, and the distance between the two. This distance changes as the diaphragm move in response to sound. If the distance changes, current flows in the wire as the battery maintains the correct charge. The amount of current is essentially proportional to the displacement of the diaphragm and is so small that it must be electrically amplified before it leaves the microphone.

A common variant of this design uses a material with a permanently imprinted charge for the diaphragm. Such a material is called an electret and is usually a kind of plastic. Plastic is a pretty good material for making diaphragms since it can be dependably produced to exact specifications. The major disadvantage of electrets is that they lose their charge after a few years and cease to work.

III.4 Signal Processing Technique Involved

III.4.a Sampling Theorem

In today engineering practices, computer increasingly used to perform the signal processing applications or called digital signal processing. In the digital signal processing, it is important to spend more time examining issues of sampling [16]. The sampling theorem stated that a bandlimited signal can be reconstructed exactly if it is sampled at a rate at least twice the maximum frequency component in it. Figure 9 illustrates the spectrum of bandlimited signal *g (t)*.

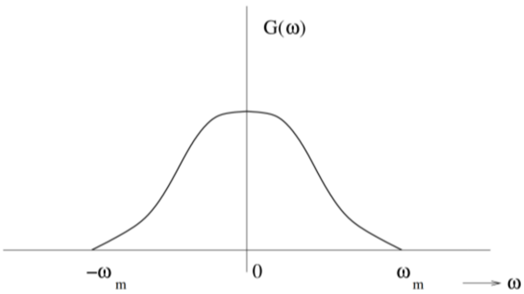


Figure 9. The spectrum of bandlimited signal g(t) [16]**.**

The maximum frequency component of *g (t)* is f_m. To recover the signal *g (t)* exactly from its samples it must be sampled at rate $f_s \geq 2f_m$. The minimum required sampling rate is called Nyquist rate (18).

$$f_s = 2 f_m \tag{18}$$

III.4.b Fast Fourier Transform (FFT)

From the algorithmic point-of-view, the fast Fourier transformation is the optimization of the discrete Fourier transformation (DFT) which reduce the time complexity of the algorithm from $O(N^2)$ to $O(N \log N)$ [17]. It is a simple computational trick that JW Cooley and John Tukey outlined in their classic 1965 paper [17].

An original DFT has a forward and inverse form which is defined as follows in (19) and (20)

<u>Forward Discrete Fourier Transform (DFT)</u>

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \tag{19}$$

<u>Inverse Discrete Fourier Transform (IDFT)</u>

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N} \tag{20}$$

The transformation from $x_n \rightarrow X_k$ is a translation from the spatial domain to frequency domain. It can be very useful in both exploring the power spectrum of a signal, and for transforming certain problems for more efficient computation.

$$\vec{X} = M \cdot \vec{x}$$

with the matrix *M* given, by

$$M_{kn} = e^{-i2\pi kn/N}$$

The computation cost of this straightforward procedure is $O(N^2)$. It comes from the computation cost from the calculation of M_{kn} and the matrix-vector multiplication of x.

To reduce the computational cost of the DFT, we can explore the symmetry of the problem. Cooley and Tukey show analytically the symmetry of this problem and results in a more cost efficient algorithm. The equation (20) shows the value of X_{N+k} from (19) where we used the identity $exp(i2\pi n) = 1$ which holds for any integer n.

$$X_{k+N} = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi(k+N)n/N}$$

$$X_{k+N} = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi n} \cdot e^{-i2\pi kn/N}$$

$$X_{k+N} = \sum_{n=0}^{N-1} x_n \cdot (1) \cdot e^{-i2\pi kn/N}$$

$$X_{k+N} = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \qquad (21)$$

The equation (21) shows a nice symmetry property of the DFT:

$$X_{k+N} = X_k$$

by a simple extension, we have,

$$X_{k+i\cdot N} = X_k$$

for any integer *i*. This symmetry can be exploited to compute the DFT much more quickly.

Cooley and Tukey [17], [18] showed that it's possible to divide the DFT computation into two smaller parts. From the definition of the DFT we have (22):

$$\begin{aligned} X_k &= \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \\ X_k &= \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-i2\pi k(2m)/N} + \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-i2\pi k(2m+1)/N} \\ X_k &= \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-i2\pi km/(N/2)} + e^{-i2\pi k/N} \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-i2\pi km/(N/2)} \end{aligned} \qquad (22)$$

We've split the single Discrete Fourier transform into two terms which themselves like smaller Discrete Fourier Transforms, one on the odd-numbered values, and one on the even-numbered values. So far, however, we haven't saved any computational cycles. Each term consists of $(N/2) * N$ computations, for a total of N^2.

The trick comes in making use of symmetries in each of these terms. Because of the range of k is $0 \le k < N$, while the range of n is $0 \le n < M \equiv N/2$, we see from the symmetry properties above that we need only perform half the computations for each sub-problem. Our $O(N^2)$ computation has become $O(M^2)$, with M half the size of N.

But there's no reason to stop there: if our smaller Fourier transforms have an even-valued M, we can reapply this divide-and-conquer approach, halving the computational cost each time, until our arrays are small enough that the strategy is no longer beneficial. In the asymptotic limit, this recursive approach scales as *O (N log N)*.

III.4.c Butterworth Filter

The Butterworth filter is a type of signal processing filter designed to have a flat frequency response as possible (no ripples) in the pass-band and zero rolls off a response in the stop-band. Butterworth filters are one of the most commonly used digital filters in motion analysis and in audio circuits. They are fast simple to use. Since they are frequency-based, the effect of filtering can be easily understood and predicted. Choosing a cutoff frequency is easier than estimating the error involved in the raw data in the spline methods.

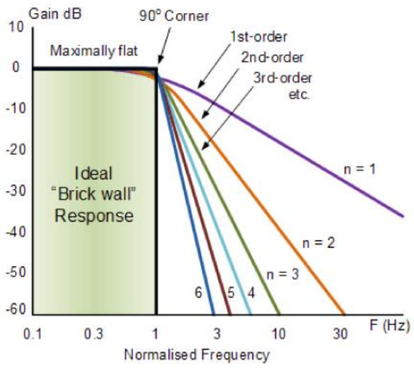


Figure 10. The low-pass Butterworth filter [19]**.**

However, one main drawback of the Butterworth filter is that it achieves this pass-band flatness at the expense of a wide transition band as the filter changes from the pass-band to the stop-band. It also has poor phase characteristics as well. The ideal frequency response referred to as a "brick wall" filter in Figure 10.

Note that the higher the Butterworth filter order, the higher the number of cascade stages there are within the filter design and the closer the filter becomes to ideal "brick wall" response. However, in practice, this "ideal" frequency response is unattainable as it produces an excessive pass-band ripple.

III.5 Envelope Fitting with Least Square Method

III.5.a Introduction

The envelope fitting is another approach which widely used to determine the damping from a free vibration curve. The approach is to fit an exponential curve passing through the peaks amplitudes, as presented in Figure 11. As already mentioned, the decay profile is the exponential part of the (11) as shown in (23).

$$x(t) = x_0 \cdot \exp(-ht) \tag{23}$$

The illustration of the envelope fitting method is shown in Figure 11.

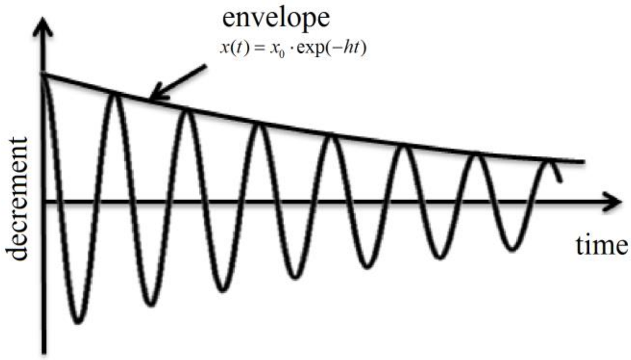


Figure 11. Envelope fitting of the transient response of an underdamped single-degree-of-freedom system [20]**.**

Though more accurate, the envelope fitting approach similar issue with the logarithmic decrement method. If the damping is not of viscous form, the fitting of the envelope along the whole transient response is likely to be of limited quality. Besides, all points from the transient response contain damping information, but bot method is used only a very small percentage of this available information, i.e. only peak data. Both methods are therefore limited in terms of efficiency. Another limitation related to both methods is that they are both strongly dependent on the sampling rate used to collect data. The lower the sampling rate is, the worse the approximation of actual amplitude is.

III.5.b Implementation of Least Square Method

For the implementation of the envelope fitting is the least square method. For this problem, we basically want to fit an exponential function in which contains noise. Linear least squares can be used to fit an exponent. However, the linear least square problem that is formed has a structure and behavior that requires some careful attention to fully understand. Usually, fitting is used because the data is noisy. If only the number of data points equal to the number of free variables in a system of equations is used, the estimate of parameters will generally be poor.

Suppose we have a differential equation for damped free vibration as shown in (11) and have solution (15). The exponential part of this solution is expressed in (23).

For a system, whose behavior can be defined by exponential decay, the parameters for the decay function can be found using least-squares. Since the data usually has measurement errors, the measured data from an exponential decay will usually contain an error term, ε_k.

$$x_k = x_0 \cdot \exp(-ht) + \varepsilon_k \tag{24}$$

Ideally, this equation could be directly set up as a linear least squares problem. However, minimizing the norm of ε_k requires solution via methods other than linear least squares. To formulate this problem as a linear least square minimization, a new error term inside the exponent is introduced, δ_k.

$$x_k = x_0 \cdot \exp(-ht_k + \delta_k) + \varepsilon_k \tag{25}$$

The usual way to set this problem up is to minimize the norm of epsilon.

$$\left\{ \begin{array}{c} \arg\min \|\varepsilon\|_2 \\ x_0, h, \varepsilon \\ s.t. \\ x_k = x_0 \cdot \exp(-ht_k + \delta_k) + \varepsilon_k, \vee k \\ \delta_k = 0, \vee k \end{array} \right\} \tag{26}$$

However, if the problem is set up to minimize the 2-norm of δ_k, then a linear least squared minimization can be formed.

$$\left\{\begin{array}{c} \arg\min_{x_0, h, \delta} \|\delta\|_2 \\ s.t. \\ x_k = x_0 \cdot \exp(-ht_k + \delta_k) + \varepsilon_k, \vee k \\ \varepsilon_k = 0, \vee k \end{array}\right\} \tag{27}$$

To linearize this problem, the terms in the constraints are rearranged, the natural log of each side is taken, and the properties of logarithms are isolated terms.

$$\left\{\begin{array}{c} \arg\min_{x_0, h, \delta} \|\delta\|_2 \\ s.t. \\ \ln(x_k - \varepsilon_k) = \ln(x_0) - ht_k + \delta_k, \vee k \\ \varepsilon_k = 0, \vee k \end{array}\right\} \tag{28}$$

The problem statement is simplified by eliminating the epsilon term and defining $a = \ln(x_0)$ and $b = -h$.

$$\left\{\begin{array}{c} \arg\min_{a, b, \delta} \|\delta\|_2 \\ s.t. \\ \ln(x_k - \varepsilon_k) = a + bt_k + \delta_k, \vee k \end{array}\right\} \tag{29}$$

Furthermore, this constrained optimization problem is restated as an unconstrained optimization problem.

$$\arg\min_{a,b} \sqrt{\sum_{k=1}^{K} (a + b \cdot t_k - \ln(y_k))^2} \tag{30}$$

Thus, the least squares can be used to solve this problem.

III.6 Signal-to-noise ratio

Summarizing from [21] signal-to-noise-ratio (SNR or S/N in short) is a measure used in instrumentation that compares the level of the desired signal to the level of noise in the background. It is defined as the ratio of signal power to the noise power, commonly in decibels. A ratio more than 1/1 or greater than 0 dB indicates more signal than noise. SNR is commonly used in the electrical signal, but it can be applied to any form of signal e.g. isotope levels in an ice core and biochemical signaling between cells.

III.6.a Definition

As previously mentioned, SNR is defined as the ratio of the power of a signal and the power of unwanted signal (noise) as shown in (31)

$$SNR = \frac{P_{signal}}{P_{noise}} \tag{31}$$

where P is the average power. Both signal and noise power must be measured at the same or equivalent state in a system and within the same system bandwidth.

III.6.b SNR in decibels

Most signals have a very wide dynamic range; signals are often expressed using the logarithmic decibel scale. From the definition of the decibel, signal and noise can be expressed in dB as (32) and (33)

$$P_{signal,dB} = 10\log_{10}(P_{signal}) \tag{32}$$

$$P_{noise,dB} = 10\log_{10}(P_{noise}) \tag{33}$$

In similar forms, SNR in decibels can be expressed as (34)

$$SNR_{dB} = 10\log_{10}(SNR) \tag{34}$$

By applying the quotient rule for logarithms

$$\begin{aligned} SNR_{dB} &= 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) \\ SNR_{dB} &= 10\log_{10}(P_{signal}) - 10\log_{10}(P_{noise}) \end{aligned} \tag{35}$$

The last result in contains signal power and noise power in form of (32) and (33), yields an important formula for SNR in decibels as shown in (36)

$$SNR_{dB} = P_{signal,dB} - P_{noise,dB} \tag{36}$$

III.7 Hypothesis

From the known envelope fitting method for approaching exponential decay experimentally in section III.5 and currently established method in section II.1 and II.2, we propose the hypothesis as follows, as the recorded induced longitudinal stress wave follows the damped free vibration except it contains some noise, the R^2 of the envelope fitting of the longitudinal stress wave signal can be used for indicating whether the recorded signal is a good representation of the longitudinal stress wave signal, or it contains a large portion of signal so that we need to repeat and recheck the recording procedure and measurement setup.

CHAPTER IV
RESEARCH METHODOLOGY

IV.1 Research Locations

This research conducted in various places. The design and implementation of the instrument, several instrument validation tests are conducted in Laboratory of Sensor and Tele-Control System at the Department of Nuclear Engineering and Engineering Physics. The microphone test is conducted in the Acoustic Assessment Studio at the same Department. Wood longitudinal stress wave data are collected from the Laboratory of Structure at the Department of Civil and Environmental Engineering.

IV.2 Tools and Materials

We used simple and price affordable materials, i.e. single board computer, microphone, USB sound card, and 3.5" resistive touchscreen display to implement a semi-automatic longitudinal stress wave technique instrument. The GW INSTEK GOS-620 oscilloscope and function generator used in one of the instrument validation steps. The Behringer TRUTH B1030A flat frequency response speaker and Microsoft Surface Pro 4 with Windows 10 operating system and Frequency Generator software by Peter Fares are used in microphone test. The detailed function of the tools and materials will be covered in the later section Research Method section. We designed the instrument to be able to build-by-ourselves so that people can reproduce this instrument easily.

For enhancing the nondestructive evaluation method, we derived some analysis from the raw data of research by [7], [8]. They are both researched same wood. The data contains six different kinds of wood with each type of wood consists of 10 samples and for each sample three longitudinal stress waves is taken. The data contains recorded signal for the following wood, viz, Sonokeling, Teak, Sukun, Acacia, Munggur, and Mahogany. The complete data can be seen in Appendix.3.

IV.3 Proposed Enhanced LSWM

IV.3.a Proposed Enhanced Method

Parallel with the design and implementation of the nondestructive evaluation instrument, in this thesis, we introduce enhanced LSWM. Common LSWM for measuring the MOE of wood is shown in Figure 12.

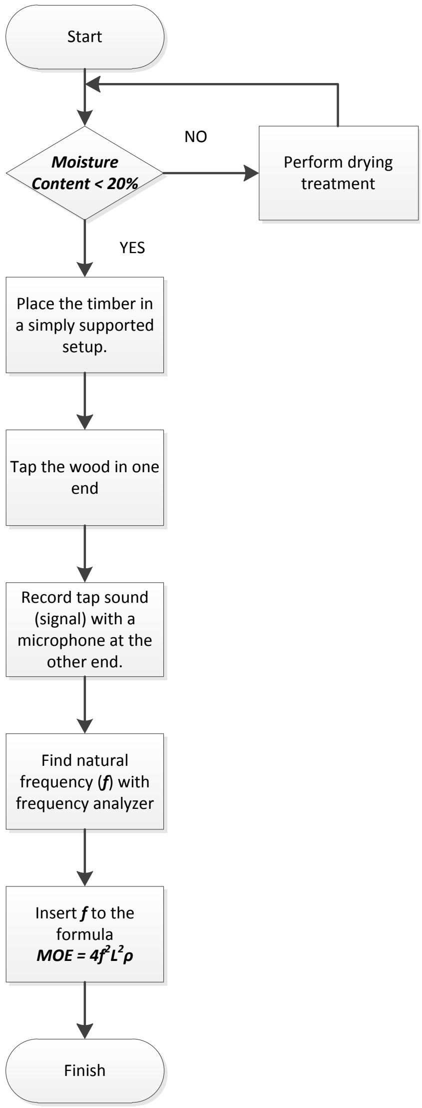


Figure 12. Current well-known LSWM for predicting the MOE of wood (summarizing from [8]**).**

The measurement starts with checking the moisture content of the wood, if we confident that the moisture content of the wood is below 20% or we want to neglect the moisture content effect in the measurement, the measurement can be started. It is empirically known that moisture conditions in wood of less than 20% will not allow decay to occur [6].

First, we place the wood in simply supported setup and place the recording tool in one end and tap the wood at the other end. The measurement done in near field area for minimizing reflection and absorption which might occur in far field area. After tapping finished and recorded, we can analyze the frequency spectrum with a frequency analyzer and get the estimated natural frequency. Then, the estimated natural frequency is passed to the dynamic MOE formula as shown in equation (1). In contrast, the proposed enhanced method is shown in Figure 13. This method adds additional validation step to the longitudinal stress wave signal. This way, we can have a measurement about how much we confident with the result. The proposed enhanced LSWM is the basis of our instrument.

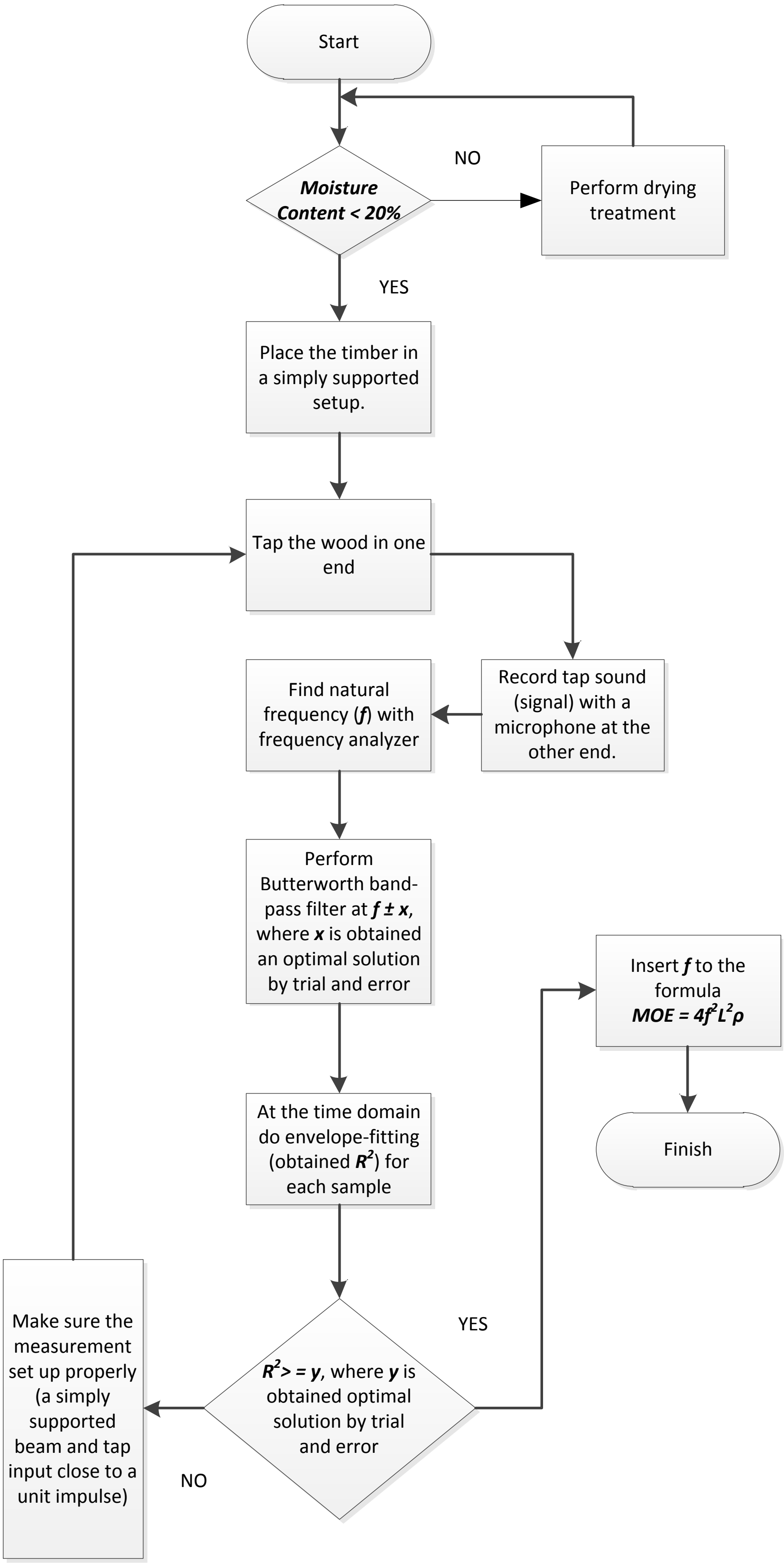


Figure 13. Proposed enhanced LSWM.

IV.3.b Pre-research of the Enhanced LSWM Implementation

Figure 14 is one of the spectra of longitudinal stress wave signal of Mahogany woods recorded by [7] and [8]. The Mahogany wood is the wood with highest MOE_s vs MOE_d correlation (R^2) in their observation. We can see that the signal is not a pure frequency signal. It is a multiple-frequency signal. From the figure, we know that the estimated natural frequency is 1.48 kHz with magnitude -76.58 dB (estimated natural frequency is the lowest frequency with the highest magnitude within range 800-4100 Hz).

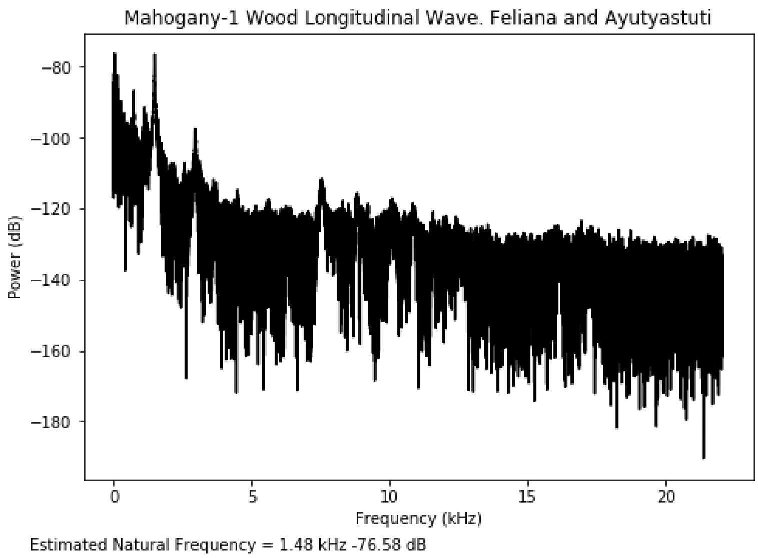


Figure 14. The Mahogany #1 sample of Feliana's work [7] **(the image is constructed by us in this thesis based on her data).**

As discussed in IV.3.a, we perform a band pass filter after obtaining the estimated natural frequency. In this section, as a pre-research result, we will perform a band pass filter with cut-off frequency of *estimated natural frequency* ± *200* Hz and 3rd order Butterworth band-pass filter. The filtered signal is shown in Figure 15.

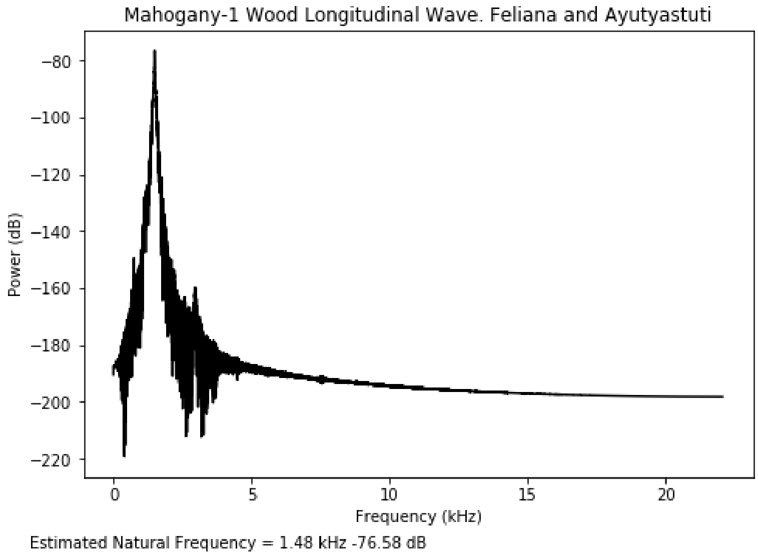


Figure 15. The filtered Mahogany-1 at ± *200* Hz from estimated natural frequency with 3rd order Butterworth filter.

We can see the exponential decay of the signal in Figure 15 in the spatial domain as shown in Figure 16

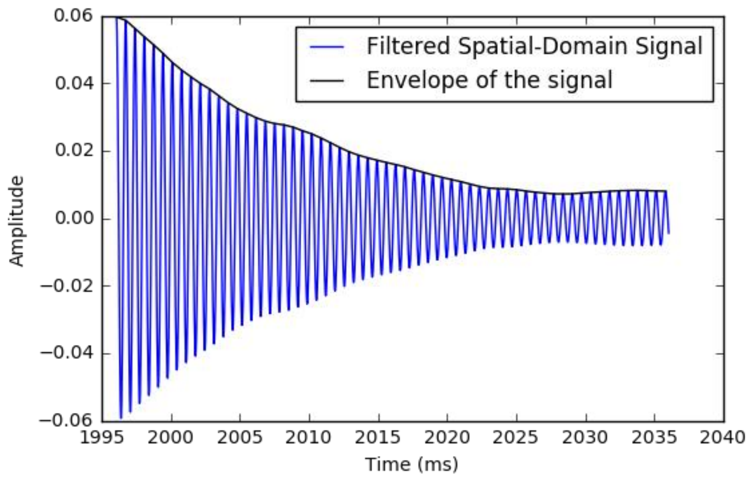


Figure 16. The filtered Mahogany #1 in the spatial domain.

We only take 1764 (this number is 1/25 of sampling rate) data points of the signal started from the highest magnitude. We derived the envelope of the signal using a naïve implementation of the envelope detector. The envelope detector basically taking each peak for every cycle of the oscillation. It is a data point in which has the previous and next data lower than it value. Thus, only the peak of the sinusoid signal that can be passed. An example of the implementation is shown in code snippet below.

```
# y is the filtered signal with band pass filter
# y_envel is y-axis value of envelope
# t_envel is x-axis value of envelope
# len is function to calculate the length of a list
y_envel = [y[data] for data in range(len(y)-1)
          if (y[data-1]<y[data] and y[data+1]<y[data])]
t_envel= [data for data in range(len(y)-1)
          if (y[data-1]<y[data] and y[data+1]<y[data])]
```

After obtaining the envelope of the signal, we can perform the envelope fitting procedure. Figure 17 shows the envelope fitting of the Mahogany-1 sample. Some bumps in the envelope is normal, since we filter a signal with multiple frequency on a limited range as previously mentioned.

From the Table 1, the Mahogany wood is a wood with the highest MOE_s vs MOE_d correlation (R^2) based on [7], [8]. It is not surprising if the envelope of the filtered signal of the Mahogany wood follows the exponential form.

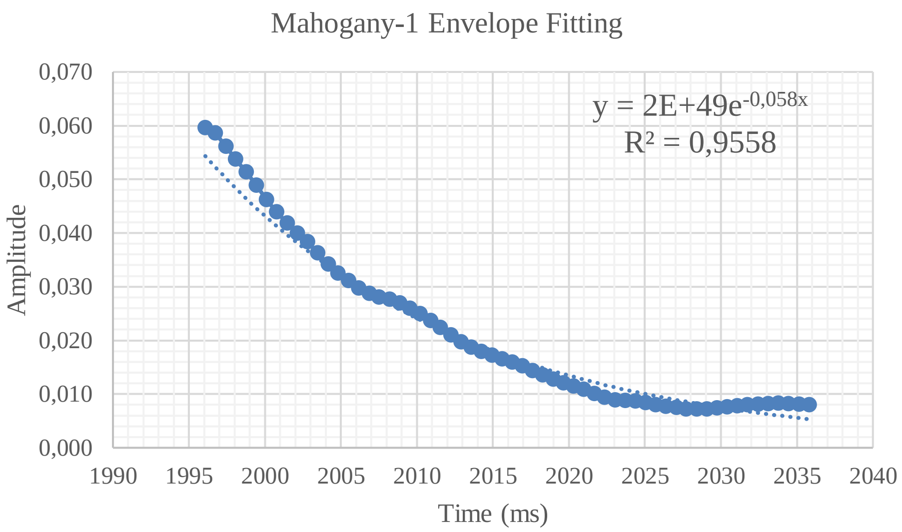
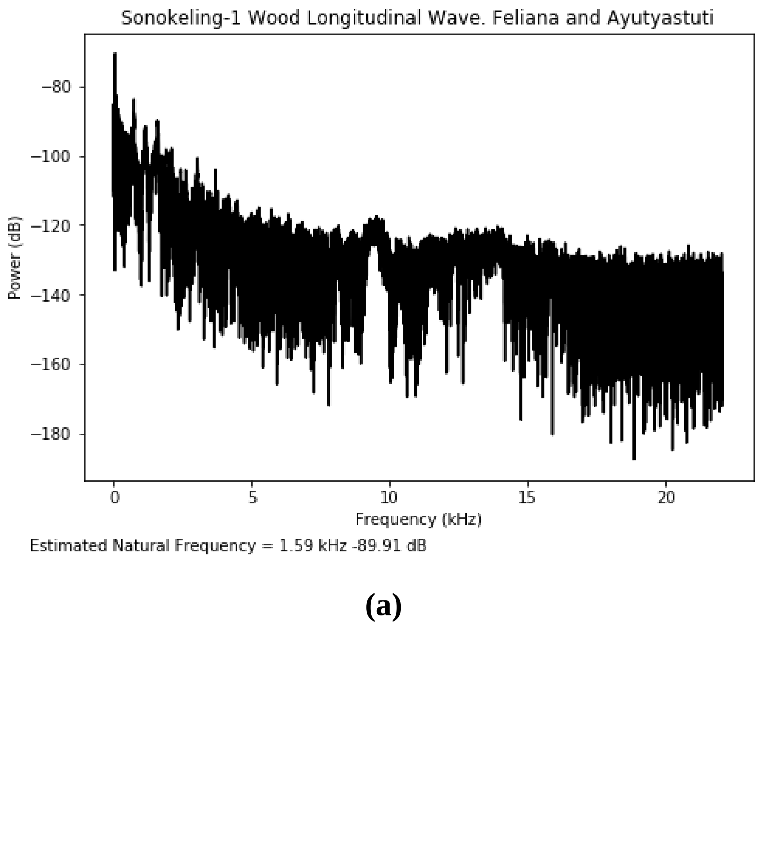


Figure 17. The envelope fitting result of the Mahogany #1 wood.

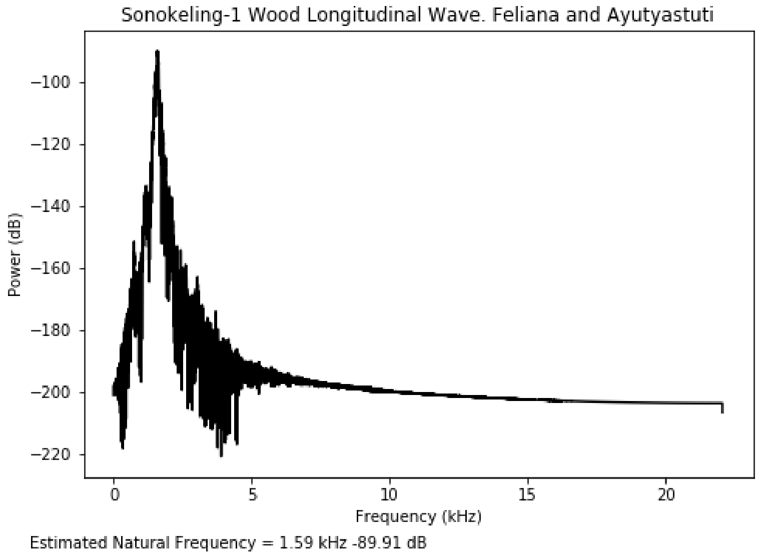
Besides the R^2 of the envelope fitting, it produces the damping ratio information. From the relationship on (11), the estimated damping factor, *h*, is the exponential component in fitting equation, such that, for Mahogany #1 is 0.058. We know that the estimated natural frequency of this wood is 1.48 kHz, we can get the damping ratio by divided damping factor by estimated fundamental natural frequency and get the result of 3.919×10^{-5}.

If we repeat the procedure for the Sonokeling wood which has the poorest MOE_s vs MOE_d correlation (R^2) based on [8], [7], we can get the signal as shown Figure 18. We can see that the Sonokeling's signal is noisy and hard to interpret.

If we perform the same procedure to this signal, we can get the envelope fitting result on Figure 19. The R^2 of the envelope fitting for this signal is only 0.742, which is smaller than Mahogany-1. Visually, we can see several bumps which might the result of unexpected noise either from background noise or some errors during measurement. In this case, the calculated damping ratio from the fitting equation also useless because the signal is not following the exponential model as expected.

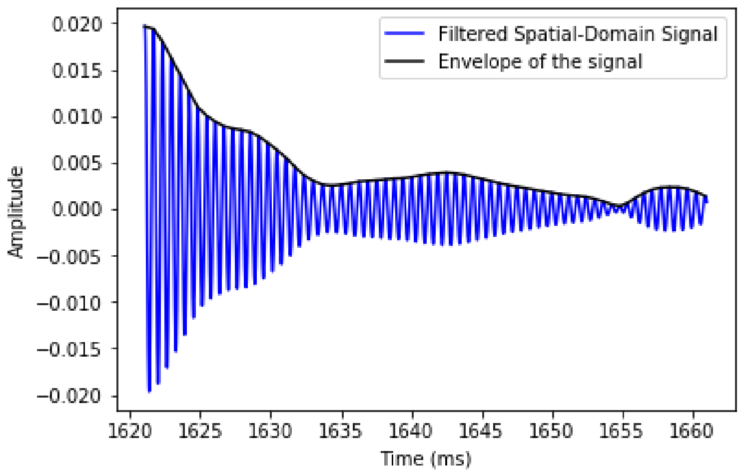


(a)

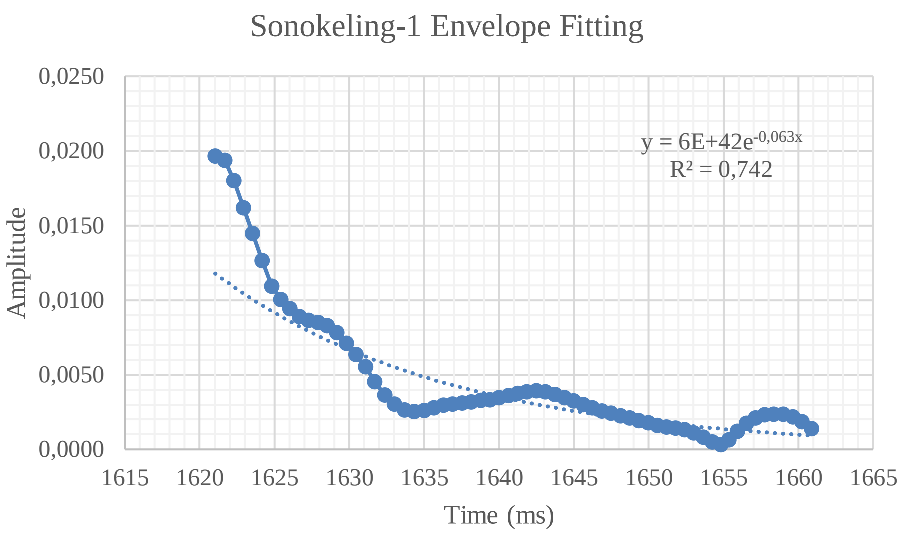


(b)

Figure 18. Longitudinal stress wave signal of Sonokeling #1 sample before filtering (a), and after band-pass filtered in estimated natural frequency (b).



(a)



(b)

Figure 19. Exponential decay of impulse response (wood's response when tapped by an impactor) after band-pass filtered in estimated natural frequency. For Sonokeling, the exponential decay is not smooth (a) and yields poor envelope fitting (b).

The data we analyzed is the same data from the Feliana's and Ayutyastuti's paper. From this observation, we can create an initial conclusion that if the R^2 of the envelope fitting shows a poor correlation between the filtered signal and the exponential model, there is a high indication that we have a high level of background noise or we didn't set up the measurement properly. Hence, we have a better understanding about the recorded longitudinal stress wave decided whether to continue the MOE evaluation or fix the measurement setup and redo the measurement.

IV.4 Instrument Design

IV.4.a Behavioral Design of Instrument

In a nutshell, in this research, we created an audio spectrum analyzer. The most important specification for the audio spectrum analyzer is the frequency range. For this research, we take the frequency range from previous work done in [8]. The comparison of the frequency range studied is shown in Figure 20. From the figure, the minimum and the maximum frequencies are 1041.90 Hz and 1913.30 Hz. We also considering the natural frequency of the steel which also studied by her and has the minimum and the maximum frequency of 3953.90 Hz and 3970.80 Hz. Thus, we defined the frequency response specifications for the instrument as 800 – 4100 Hz. We lower the lower limit and increase the upper limit as a buffer and make us confident that natural frequency of wood is under our instrument's capability. We expect the instrument to be robust, easy to use and mobile.

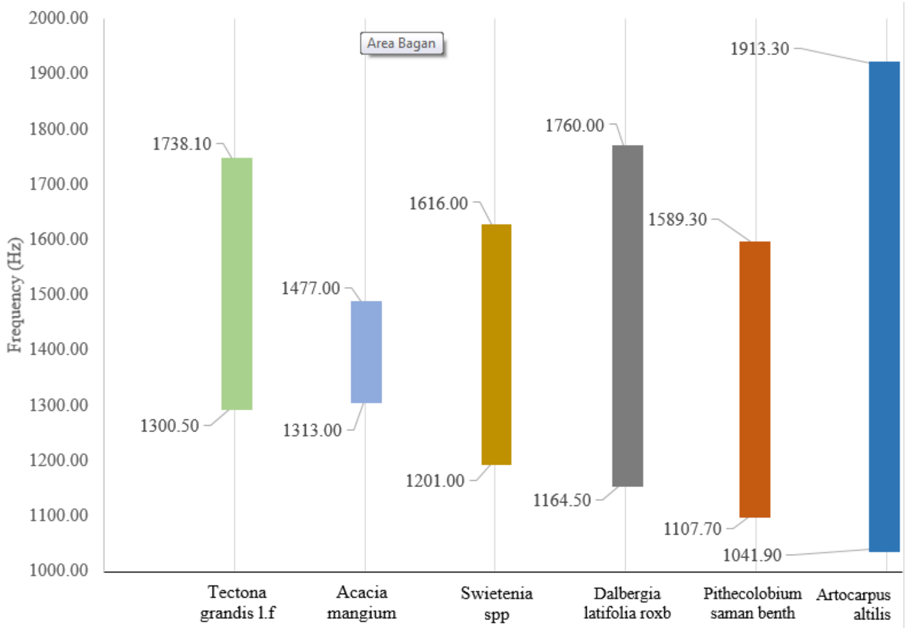


Figure 20. The frequency ranges of various wood observed in [8]**.**

IV.4.b The Structural Design of Instrument

In this research, there are two kinds of the wood parameter that will be predicted, i.e. the MOE and the envelope fitting result of longitudinal stress wave. In the design of the instrument, we proposed a semi-automatic instrument that records the longitudinal stress wave and processes the data signal to yield the MOE and the damping ratio of the longitudinal stress wave of wood.

Generally, in the laboratory, people measuring the longitudinal natural frequency of wood by using a personal computer, either desktop or laptop. Then, they are using built-in FFT module embedded in the software, i.e. MATLAB, Scilab, etc. For maintaining the robustness and the mobility of the instrument, we design the instrument to be able to calculate the NDE results directly on the instrument's display. Figure 21 shows a complete hardware structural design of the instrument.

In this research, for the sensing element, we used a dynamic microphone which has a frequency range of 100 – 5000 Hz claimed by the manufacturer. We rechecked the microphone frequency range and will discuss it in the next section. We used USB soundcard as the analog-to-digital converter to sample the signal.

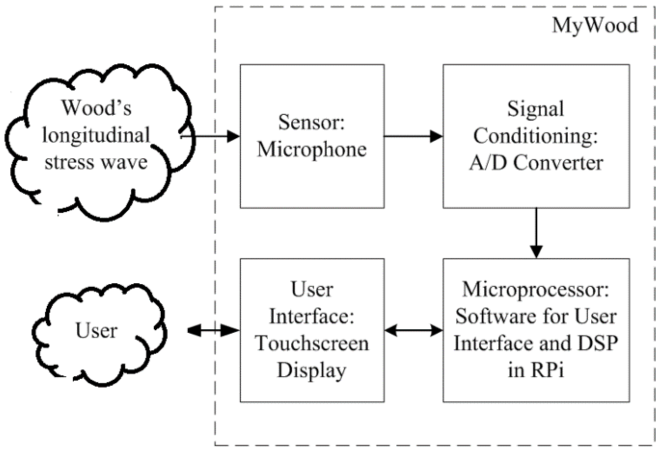


Figure 21. The proposed instrumentation system design.

For the processing unit, we used a single board computer called Raspberry Pi 2 (RPi). It embedded with quad-core Broadcom BCM2835 900 MHz CPU, it is equipped with 4-USB ports and some embedded communications port i.e. SPI, I2C, and UART. One of the most interesting facts is it only consume around 5 W of power. We used Raspbian Linux as its operating system.

For the user interface, we used a 3.5-inch resistive touchscreen display to make it possible user interaction without a keypad. We used the N-tier application architecture for implementing the software for digital signal processing and user interface on the Raspberry Pi. The N-tier application architecture enables us to reuse majority of code in business/logic layer in another presentation layer. A complete diagram of the architecture is shown in Figure 22.

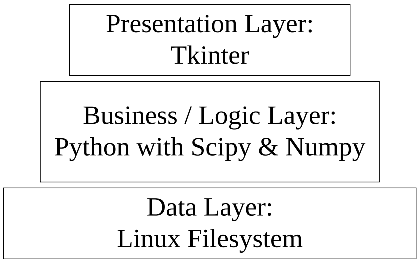


Figure 22. The N-tier application architecture for the software implementation.

We used Linux file system to store data. There are no special data stored in this software, only recorded audio file stored. For the business or logic layer, we used Python programming language with Scipy and Numpy library embedded. For the presentation layer, a graphical user interfaces library Tkinter, and graph plotting library Matplotlib used in this research.

IV.4.c Implementation of Longitudinal Stress Wave Validation Method

After the fundamental natural frequency of longitudinal stress wave of wood is obtained, the next step is to use the f_0 to validate the recorded longitudinal stress wave of wood. The complete algorithm to this step is shown in Table 4.

Table 4. Algorithm to validate the recorded longitudinal stress wave.

1. **Filter the signal w/ with f_0 as center frequency**
2. **Show the spectrum of filtered signal (optional)**
3. **Extract envelope of LSW impulse response**
4. **Perform fitting to envelope to impulse response**
5. **Calculate the R^2 of the envelope fitting (R^2_{envl})**

To be able to perform the algorithm, in this research we perform an analysis to determine best the cut off frequency to perform signal filtering in step 1 and the order of Butterworth bandpass filter used in the implementation.

IV.5 Instrument Implementation and Testing

IV.5.a How to navigate the instrument

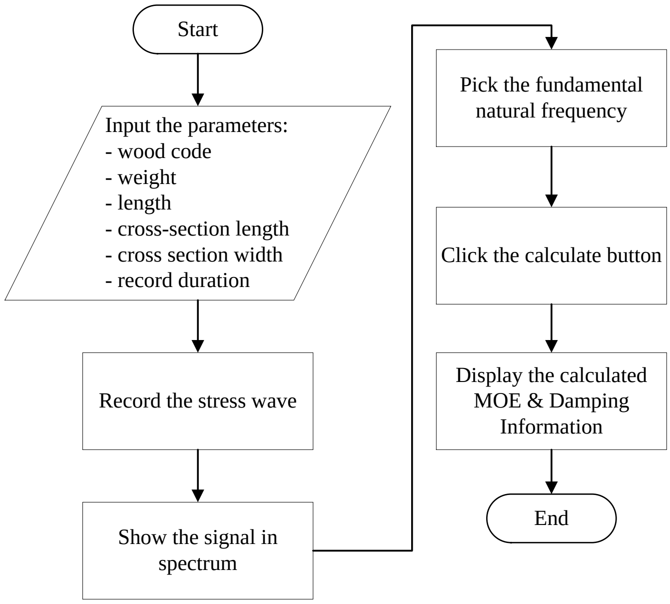


Figure 23. Flowchart of how to navigate the instrument.

The instrument is designed to be semi-automatic. The complete flowchart (scenario) of how to navigate the instrument is shown in Figure 23. When the program starts, it loads a default configuration. The user should input the parameters regarding the wood sample, including wood code, weight, length, cross-section length, cross section width, and record duration. After the parameters inputted properly the user can record the stress wave signal. The signal in time-domain automatically appears when recording stops. Next, the user can see the signal in frequency domain and pick the fundamental frequency. The fundamental frequency picked used in the calculation of the MOE using equation (1). The fundamental natural frequency is the natural frequency which has the highest magnitude, but the lowest frequency.

IV.5.b The user interface

Figure 24 shows the main view of the instrument user interface. The user can use this display with the touchscreen provided and touch the button and navigate the pointer. Basically, there are four main parts of the main view i.e. the plot area, plot toolbar, sidebar tool, and pointer position status. The plot area will display the necessary plot e.g. recording result and FFT result. The plot toolbar will help the user navigate the plot e.g. for zooming, back to original view and save the image. The sidebar tool will be the main navigation for the user, there are several buttons i.e. setting, record, FFT, pick ***f***, verify, calculate, and open. The pointer position status will show the current position of the cursor.

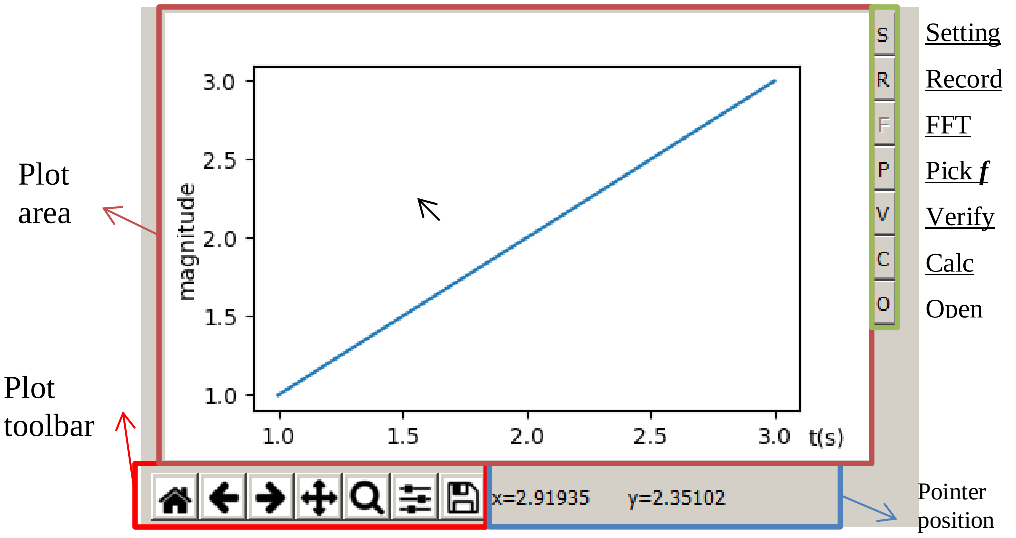


Figure 24. The main view of the instrument user interface.

IV.5.c Testing

Basically, for the validation steps, there are several tests to verify the performance of the instrument i.e. soundcard performance test, microphone performance test, FFT validation, and optimization, and attenuation information collection. The detailed explanation for each test will be discussed in section IV.6.

IV.6 Data Collection

For the test and validation, we conducted four performance tests for the microphone, the soundcard, and the FFT algorithm. The order of the test is the soundcard, the microphone, and the FFT algorithm. The soundcard test tested in the first place because we used the soundcard to test the microphone.

IV.6.a Soundcard Performance Test

The soundcard test conducted to confirm its performance to sample the audio signal. A test setup is shown in Figure 25. In this test, we generate a predefined signal with a function generator, see the waveform with an oscilloscope, and transmit the signal with audio cable to the sound card of the instruments. To cut the testing time we used a laptop computer to run remote desktop via Ethernet cable and running Audacity program remotely.

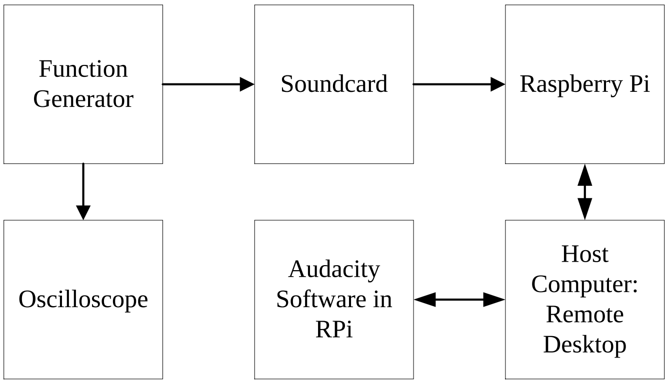


Figure 25. The soundcard performance test setup.

In this test, we test the sound with sine signal, square signal, and triangle signal and see the signal read by the soundcard. Relative sensitivity of the soundcard is also studied.

IV.6.b Microphone Performance Test

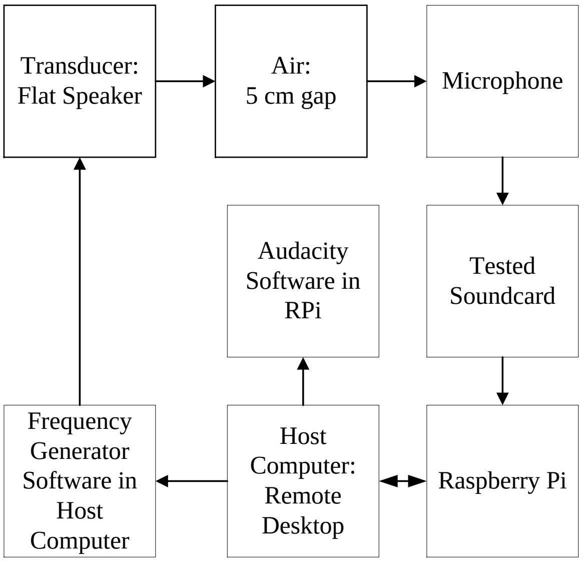


Figure 26. The microphone performance test setup.

The microphone test conducted to confirm its frequency range. A complete test setup is shown in Figure 26. The best way to test the characteristic of the microphone is using a tuning fork, because of unavailability of the tuning fork; the flat frequency (see Appendix 3.H for full specifications) speaker is used. To minimize the background noise, the test conducted in the acoustic studio which has maximum background noise of -55.6 dBFS at the time we conducted the test.

A personal computer with frequency generator software installed is used to generate the predefined signal. Then, the signal is transmitted to the flat speaker with an audio cable. A 5-cm gap from the center of the speaker to the microphone is arranged. The microphone is connected to the soundcard that has tested in the previous test setup. A remote desktop is utilized to simplified the experiment and cut the testing time.

IV.6.c Fast Fourier Transform Validation and Optimization

The third test is the FFT test. The test setup test for the FFT test is using two type of test file. The pure frequency test file and the intermodulation distortion test file. The test file is a WAV file generated with IMD Test Tool from Audio Check [22].

Due to the limitation of the microprocessor which has 900 MHz Quad-Core ARM Processor, the time complexity of the FFT implementation is very important. Thus, we study the various implementation of FFT in Python programming language and pick the best implementation. In this thesis, we compare the naïve DFT implementation, the naïve FFT implementation, the vectorized FFT implementation and Numpy FFT implementation.

IV.6.d Attenuation Characteristics Analysis

For the attenuation characteristics, we calculate the R^2 of envelope fitting of each recorded LSW collected with the method discussed in section IV.3.a. The data is gathered from previous research done in the Laboratory of Structure Engineering [7], [8]. The data contains recorded signal for the following wood, viz, Sonokeling, Teak, Sukun, Acacia, Munggur, and Mahogany.

With statistical tools, we will analyze the correlation between the damping information with the R^2 of the correlation between MOE_d and MOE_s. As mentioned in the hypothesis, we want to know whether the R^2 of envelope fitting influences the poor correlation in the LSWM prediction.

CHAPTER V
RESULT AND DISCUSSION

V.1 Physical Implementation of the Instrument

In a brief, the instrument used a dynamic microphone with a frequency range of 100 – 5000 Hz. It used a USB sound card to read the audio signal from the microphone, process the FFT in Raspberry Pi 2 Model B computer and display the result and user interface in 3.5-inch resistive touchscreen display. For the power supply, a power bank with 12000 mAh used to power the instruments. It supplied maximum 2.1A to the instrument. It has a 12×8×4 cm – a relatively small dimension. An image of the instrument is shown in Figure 27.

Figure 27. An instrument created in this research.

V.2 The Instrument's User Interface

V.2.a The Main Display

The main display of the instrument shows a set of tool to navigate the instrument. In the right side of the display, there are some navigation button with the following details

Table 5. Navigation Button and its Functions

Button	Name	Function
S	Setting	Open a setting window
R	Record	Start LSW recording
F	Frequency View	View Frequency Spectrum
P	Get Peak	Get Natural Frequency
V	Verified	Perform LSW Verification
C	Calculate	Calculate MOE_d based on setting data
O	Open	Open a recorded file

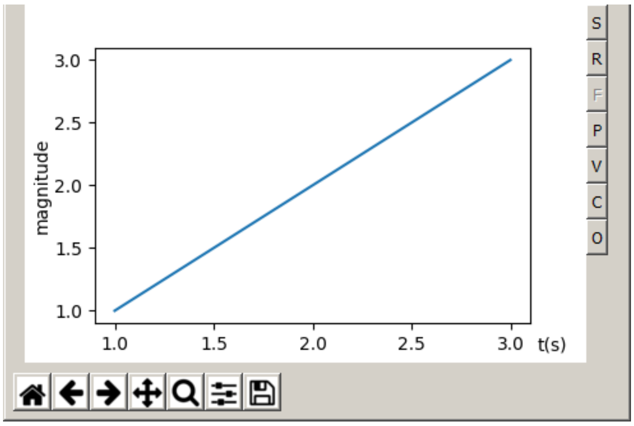


Figure 28. The main display of the instrument

V.2.b The Settings Window

The settings window provides a way to input information about wood we want to evaluate including weight, length, thickness (cross section length and width), and wood name or code. It is activated if we click the **S** button. Additionally, we can set up the duration of the recording in this window. For natural frequency field, it is automatically filled by the instrument if we use the **P** button in the main display.

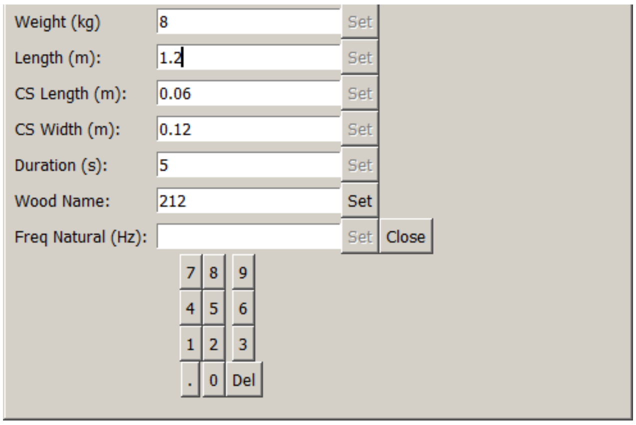


Figure 29. Settings window of the instrument

V.2.c The Recording & Open a File Result

There are two ways to evaluate a longitudinal stress wave of wood. We can record the signal directly from wood or we can open a previously recorded longitudinal stress wave. We can record the signal by clicking the **R** button and open a file in the instrument by clicking **O** button. Either way, it will show a signal in time domain right after the signal successfully recorded or opened.

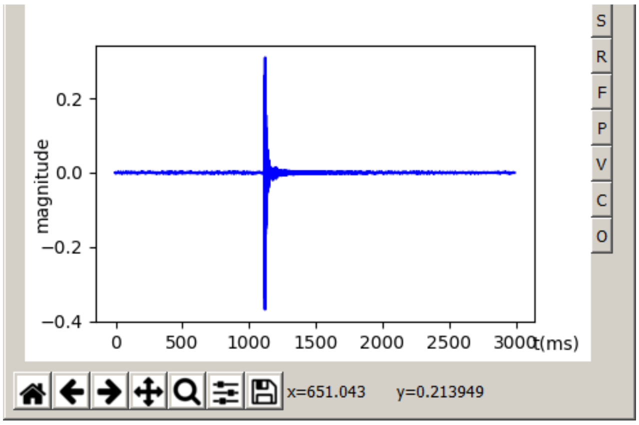


Figure 30. Screen after recording or open a previously recorded longitudinal stress wave.

V.2.d The FFT Result

We can see the frequency spectra of the signal by clicking the **F** button, and it will automatically switch **F** button to **T** button which show the signal in time domain. It is the mechanism to easily switch from frequency view to time domain view.

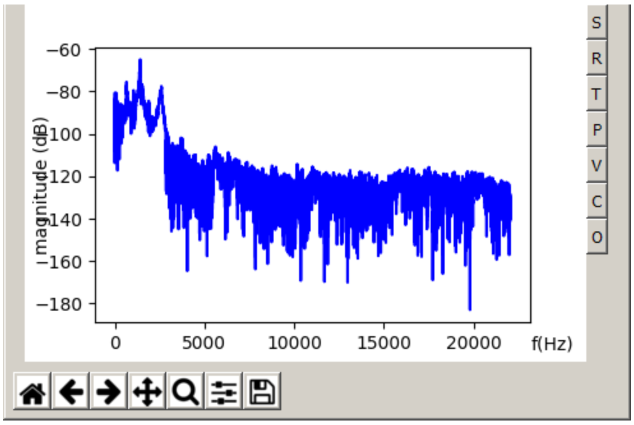


Figure 31. Frequency domain view of a longitudinal stress wave signal.

V.2.e Auto Detect Natural frequency

We provide a way to automatically detect the peak magnitude or the estimated fundamental natural frequency automatically. User can click **P** to automatically detect the peak of the signal. It is basically done by limiting the range we want to check to the range of longitudinal stress wave between 800 Hz to 4100 Hz and find the argmax of magnitude.

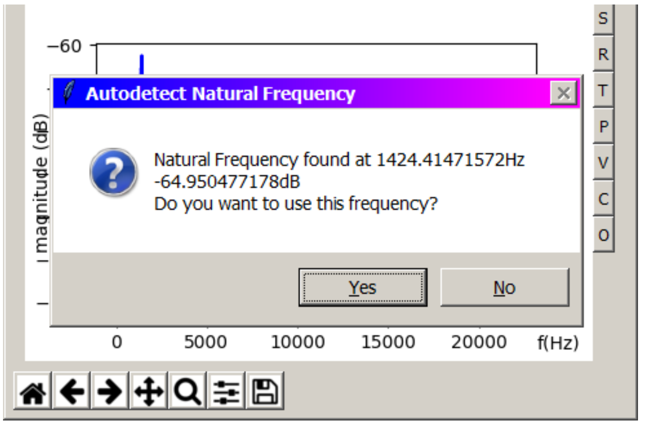


Figure 32. The natural frequency estimation prompt after the user click the P button.

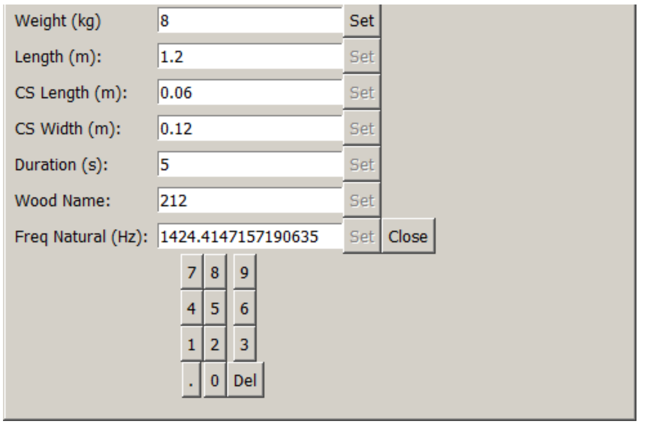


Figure 33. The frequency natural field is automatically filled if we press OK to the estimation prompt.

V.2.f The Verification Steps

We can initiate a verification steps by clicking the **V** button. It will trigger a prompt which describe some succeeding steps.

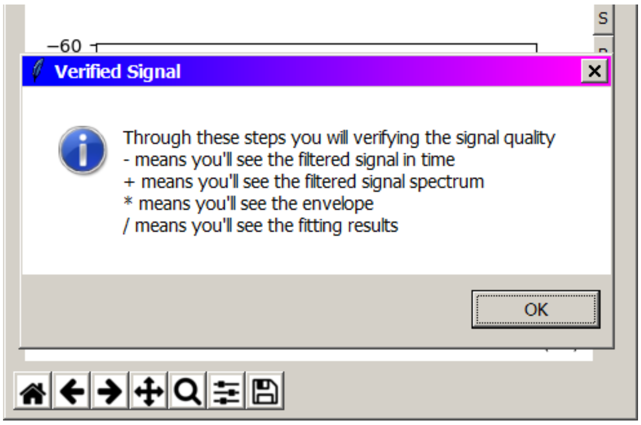


Figure 34. Direction for verification step which show immediately after clicking the V button.

After clicking OK, we can continue to the following steps.

1) Show the filtered signal (the – button)

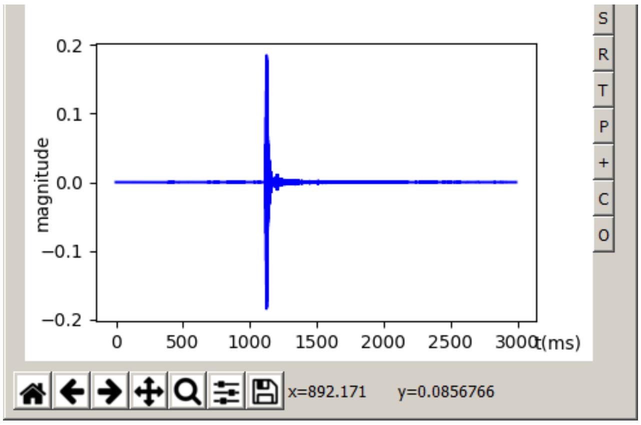


Figure 35. A resulting filtered signal of an example input.

2) Show the filtered signal spectrum (the + button)

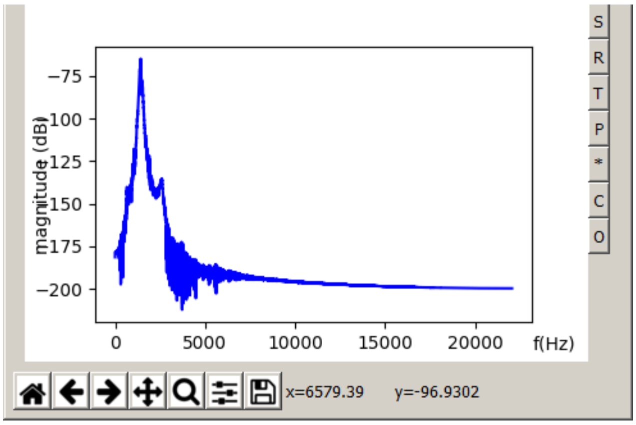


Figure 36. The resulting filtered signal spectrum of an example signal.

3) Show the envelope of filtered signal (the * button)

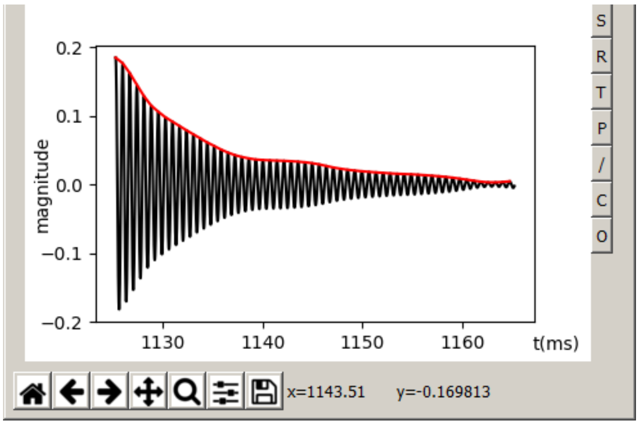


Figure 37. Resulting envelope of longitudinal stress wave (red line) and original longitudinal stress wave (black line)

4) Show the R_{envl} (the / button)

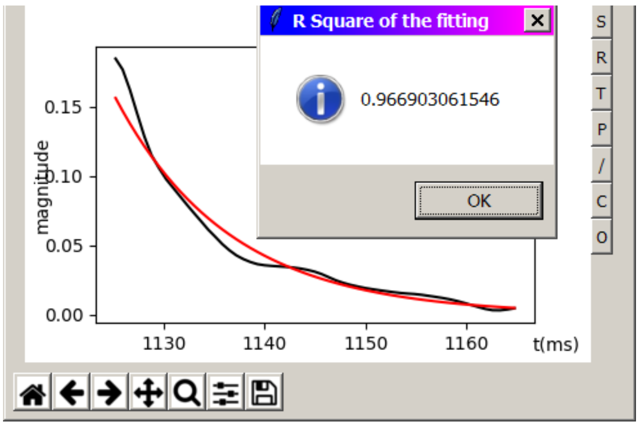


Figure 38. The prompt showing the R_{envl} and curve fitting result. Fitting result represented by red line and the original envelope is represented by black line.

V.2.g The MOE Results

The final steps of the measurement are to show the result of MOE and SNI classification. We can calculate the result by clicking the **C** button and it will immediately show the result which contains the MOE and SNI classification which derived from Table 3.

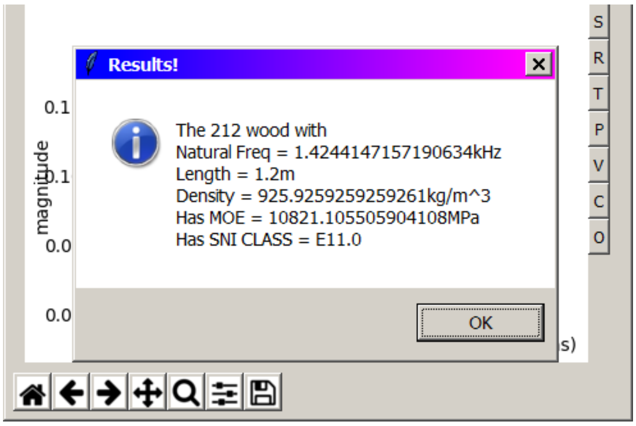
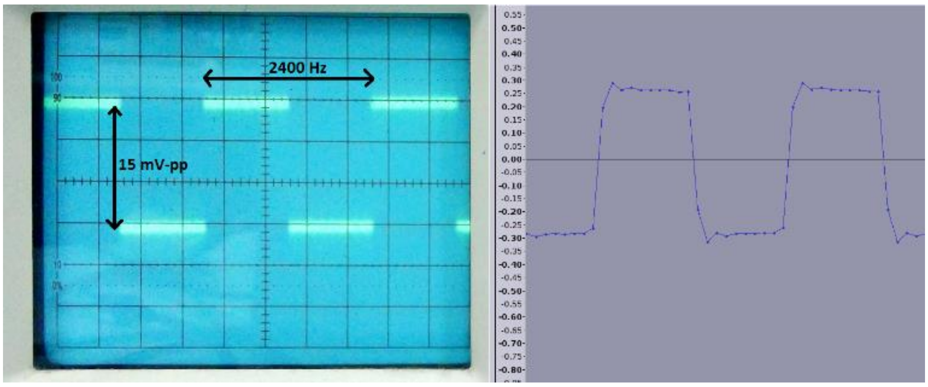


Figure 39. Final evaluation result showing the MOE and SNI classification.

V.3 Soundcard Performance Test

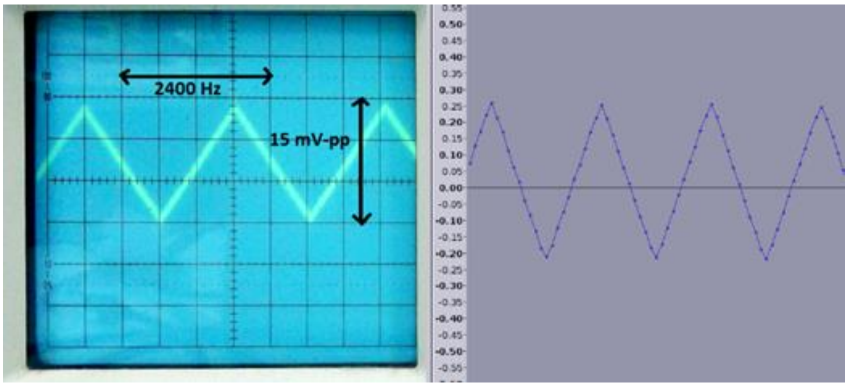
In this test, there is two test conducted, the waveform test and the frequency range test. We used a signal with magnitude 15 mV peak-to-peak throughout the test. A 48000 Hz sampling rate is used in this test. The first test is waveform test. We used 2.4 kHz signal to test the waveform.



(a) (b)

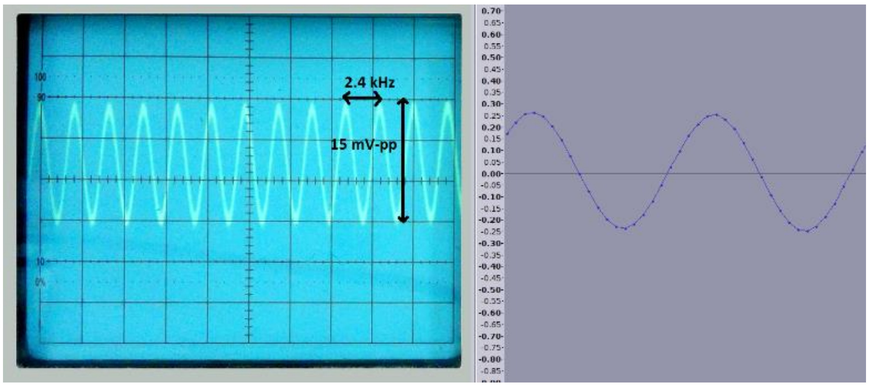
Figure 40. Square wave test, (a) is the input signal, (b) signal read by the instrument.

A square wave test is shown in Figure 40. The left side of the figure is the input signal shown in oscilloscope and the right side is the signal read by the instrument. The sound card can work well with a square wave input proved with the readings showed similar results except for a few distortions. The triangle wave test in Figure 41 shows the same result. It also works well with a triangle wave signal and the reading is almost the same with the input signal.



(a) (b)

Figure 41. Triangle wave test, (a) is the input signal, (b) signal read by the instrument.



(a) (b)

Figure 42. Sine wave test, (a) is the input signal, (b) signal read by the instrument.

The sound card works well with the sine wave signal. It reads almost the same signal with the input signal as we can see in Figure 42. After confirming the sound card able to read almost the same signal with the input signal we evaluated the frequency range from the design specification. A test from 800 Hz until 4100 Hz conducted and the result is shown in Figure 43. The soundcard performed well in the desired frequency range and have an average magnitude of -11.86 dB. The soundcard works well in the various waveforms and within desired frequency range so that there is no additional adjustment needed.

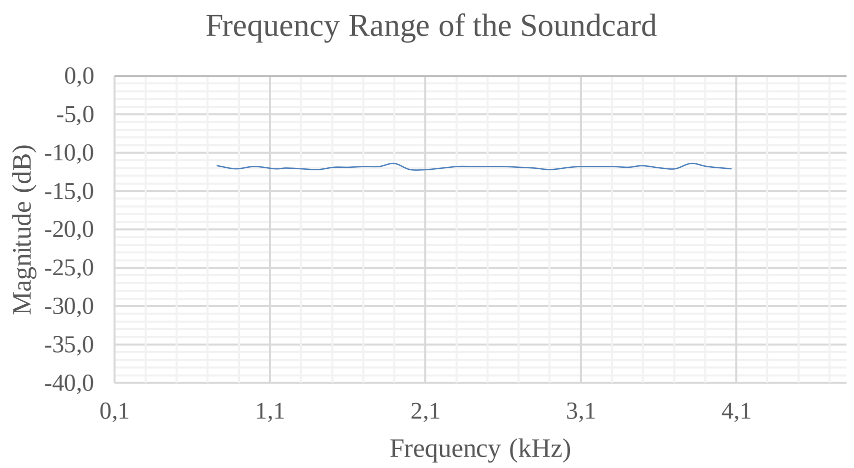


Figure 43. Soundcard response in the frequency range between 800 - 4100 Hz.

V.4 Microphone Performance Test

In the microphone performance test, we conducted frequency range test. We implement the test by testing an individual frequency from 800 Hz to 4100 Hz with 100 Hz increment. The detailed raw data is available in Appendix 4.A. The graph in Figure 44 shows the test result. Basically, the microphone able to work at the desired frequencies, but every frequency has different sensitivity. In the relatively low frequency, it is more sensitive than the opposite. We can see that the sensitivity decline from 800 Hz until 2700 Hz and raise again in the frequency that higher than 2700 Hz.

By subtracting the microphone response with the background noise, we can get the signal-to-noise ratio (SNR) in dB as shown in Figure 45. The SNR in the desired frequency range is more than 0 dB, we can say that the signal is dominant over the noise and the microphone is suitable for used in the instruments.

For the SNR graph, we can see that the graph near identical with microphone response except for relatively low frequency (below 1 kHz), the noise is higher than in the relatively high frequency so that the form is different with the microphone response.

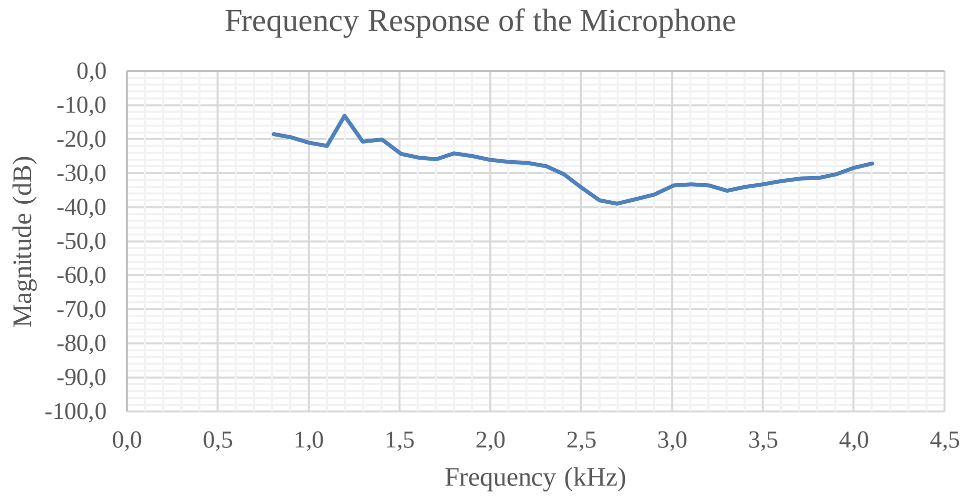


Figure 44. Microphone response in the frequency range between 800 - 4100 Hz.

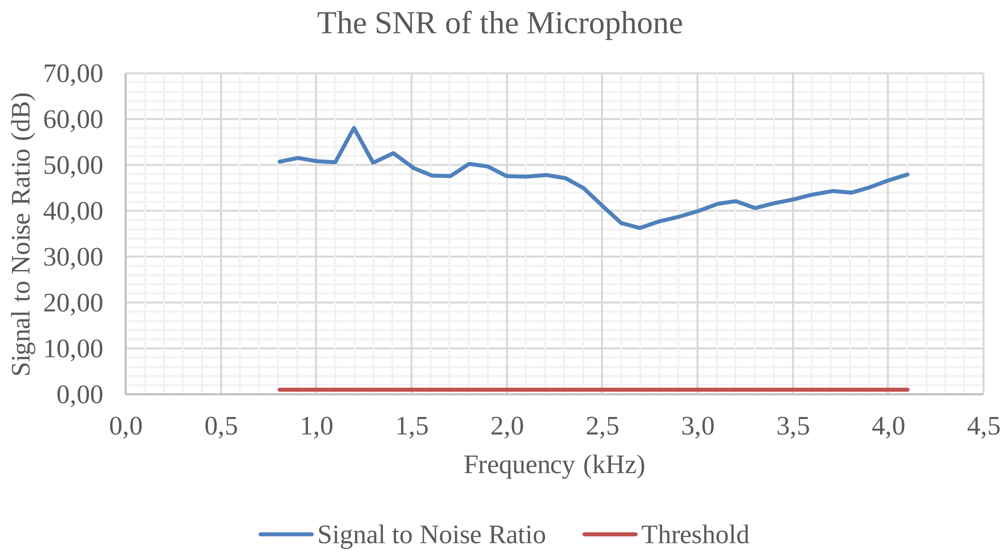


Figure 45. Signal to noise ratio of the microphone in the test.

V.5 Fast Fourier Transform Validation and Optimization

V.5.a The Naïve DFT Implementation

As mentioned in (18), the DFT expression is nothing more than a straightforward linear operation. We can compute the DFT using simple matrix multiplication as follows:

```
import numpy as np
def DFT_slow(x):
#Compute the discrete Fourier Transform of the 1D array x
    x = np.asarray(x, dtype=float)
    N = x.shape[0]
    n = np.arange(N)
    k = n.reshape((N, 1))
    M = np.exp(-2j * np.pi * k * n / N)
    return np.dot(M, x)
```

In this test, we used Python 3 Jupyter Notebook. We can double-check the result by comparing to numpy's built-in FFT function:

```
x = np.random.random(1024)
np.allclose(DFT_slow(x), np.fft.fft(x))
```

Result:

```
True
```

Just to confirm the sluggishness of our naïve implementation, we can compare the execution times of these two approaches:

```
%timeit DFT_slow(x)
%timeit np.fft.fft(x)
```

Result:

```
10 loops, best of 3: 75.4 ms per loop
10000 loops, best of 3: 25.5 µs per loop
```

Our naïve implementation over 1000 times slower than FFTPACK implementation, which is to be expected for such a simplistic implementation. But that's not the worst of it. For an input vector of length N, the FFT algorithm scales as *O (N log N)*, while our slow DFT scales as $O(N^2)$.

V.5.b The Naïve FFT Implementation

The FFT implementation involves a recursive function. This recursive algorithm can be implemented very quickly in Python, falling-back on our slow DFT code when the size of the sub-problem becomes suitably small:

```
def FFT(x):
    """A recursive implementation of the 1D Cooley-Tukey FFT"""
    x = np.asarray(x, dtype=float)
    N = x.shape[0]

    if N % 2 > 0:
        raise ValueError("size of x must be a power of 2")
    elif N <= 32:  # this cutoff should be optimized
        return DFT_slow(x)
    else:
        X_even = FFT(x[::2])
        X_odd = FFT(x[1::2])
        factor = np.exp(-2j * np.pi * np.arange(N) / N)
        return np.concatenate([X_even + factor[:N / 2] * X_odd,
                               X_even + factor[N / 2:] * X_odd])
```

We can double-check the result by comparing to numpy's built-in FFT function:

```
x = np.random.random(1024)
np.allclose(FFT(x), np.fft.fft(x))
```

Result:

```
True
```

Comparing this result with naïve DFT implementation yields:

```
%timeit DFT_slow(x)
%timeit FFT(x)
%timeit np.fft.fft(x)
```

Result:

```
10 loops, best of 3: 77.6 ms per loop
100 loops, best of 3: 4.07 ms per loop
10000 loops, best of 3: 24.7 µs per loop
```

Our calculation is faster than the naïve version by over an order of magnitude. What's more, our recursive algorithm is asymptotically *O (N log N)*. Note that we still haven't come close to the speed of the built-in FFT algorithm in numpy, and this is to be expected. The FFTPACK algorithm behind numpy's fft is a FORTRAN implementation which has received years of tweaks and optimizations. Furthermore, our Numpy solution involves both Python-stack recursions and the allocation of many temporary arrays, which adds significant computation time [17].

V.5.c The Vectorized FFT Implementation

A good strategy to speed up the code when working with Python/Numpy is to vectorized repeated computations where possible. When we talk about vectorized version, it is basically using an optimized data structure provided by Numpy in Python. Python organize a collection of number with list data structure which in this case is slower than C language implementation. We can achieve faster computation by using Numpy array, which is an array data structure which has a speed comparable to C implementation. We can do this, and in the process remove our recursive function calls, and make our Python FFT even more efficient [17].

Notice that in the previous recursive FFT implementation, at the lowest recursion level we perform N/32 identical matrix-vector products. The efficiency of our algorithm would benefit by computing these matrix-vector products all at once as a single matrix-matrix product. At each subsequent level of recursion, we also perform duplicate operations which can be vectorized. Numpy excels at this sort of operation, and we can make use of that fact to create this vectorized version of the Fast Fourier Transform:

```
def FFT_vectorized(x):
#A vectorized, non-recursive version of the Cooley-Tukey FFT
    x = np.asarray(x, dtype=float)
    N = x.shape[0]

    if np.log2(N) % 1 > 0:
        raise ValueError("size of x must be a power of 2")

# N_min here is equivalent to the stopping condition above,
# and should be a power of 2
    N_min = min(N, 32)

# Perform an O[N^2] DFT on all length-N_min sub-problems at once
    n = np.arange(N_min)
```

```
    k = n[:, None]
    M = np.exp(-2j * np.pi * n * k / N_min)
    X = np.dot(M, x.reshape((N_min, -1)))

# build-up each level of the recursive calculation all at once
    while X.shape[0] < N:
        X_even = X[:, :X.shape[1] / 2]
        X_odd = X[:, X.shape[1] / 2:]
        factor = np.exp(-1j * np.pi * np.arange(X.shape[0])
                        / X.shape[0])[:, None]
        X = np.vstack([X_even + factor * X_odd,
                       X_even - factor * X_odd])

    return X.ravel()
```

Though the algorithm is a bit opaquer, it is simply a rearrangement of the operations used in the previous recursive version with one exception: we exploit the symmetry in the factor computation and construct only half of the array. Again, we'll confirm that our function yields the correct result:

```
x = np.random.random(1024)
np.allclose(FFT_vectorized(x), np.fft.fft(x))
```

Result:

```
True
```

Because our algorithms are becoming much more efficient, we can use a larger array to compare the timings, leaving out DFT_slow:

```
x = np.random.random(1024 * 16)
%timeit FFT(x)
%timeit FFT_vectorized(x)
%timeit np.fft.fft(x)
```

Result:

```
10 loops, best of 3: 72.8 ms per loop
100 loops, best of 3: 4.11 ms per loop
1000 loops, best of 3: 505 µs per loop
```

We've improved our implementation by another order of magnitude. We're now within about a factor of 10 of the FFTPACK benchmark, using only a couple dozen lines of pure Python + Numpy. Though it's still no match computationally speaking, readability-wise the Numpy version is far superior to the FFTPACK source.

So how does FFTPACK attain this last bit of speedup? Mainly it's just a matter of detailed bookkeeping. FFTPACK spends a lot of time making sure to reuse any sub-computation that can be reused. Our numpy version still involves an excess of memory allocation and copying; in a low-level language like Fortran, it's easier to control and minimize memory use. In addition, the Cooley-Tukey algorithm can be extended to use splits of size other than 2 (what we've implemented here is known as the radix-2 Cooley-Tukey FFT). Also, other more sophisticated FFT algorithms may be used, including fundamentally distinct approaches based on convolutions (see, e.g. Bluestein's algorithm and Rader's algorithm). The combination of the above extensions and techniques can lead to very fast FFTs even on arrays whose size is not a power of two [17]. From this consideration, we used Numpy implementation to save the computational cost in the microprocessor.

V.5.d The FFT Validation

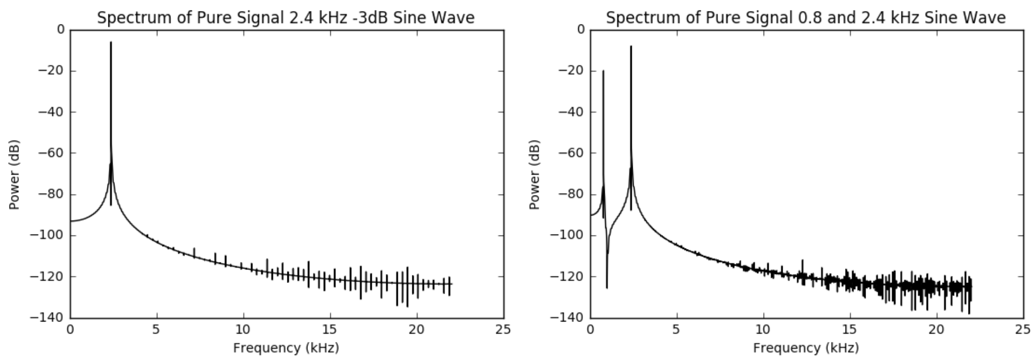
Two kinds of tests conducted to measure the performance of the implemented FFT algorithm, i.e. the pure frequency test and dual frequency test (intermodulation distortion (IMD) test). The first test is to measure the response of the implemented algorithm to a pure frequency between 100 Hz – 4100 Hz with 100 Hz increment.

The second test is to measure the level of an unwanted combination of different frequencies found in the input signal. IMD adds components to the sound that are not found in the original signal. This effect results from non-linearity in your audio system. For instance, two frequencies, f_1, and f_2 may produce new frequencies i.e. f_2-f_1, f_2+f_1, f_2-2f_1, f_2+2f_1 or any other intermodulation component [22]:

$$m \cdot f_2 \pm n \cdot f_1$$

where *m* and *n* are integers. When a complex acoustical passage is a source, the IMD can be quite extraordinary.

The first test shows that the implemented FFT algorithm is performed well with no distortion and harmonics. The example of the FFT output is shown in Figure 46 (a) the test is conducted with 3 seconds' durations wave file with -3dBFS magnitude and sample rate of the file is 44.1 kHz. There is only small harmonics and ripple in a frequency higher than 4.1 kHz.



(a) **(b)**

Figure 46. An example of FFT plot of pure sine waves at 2400 Hz (a). The FFT plot of 800Hz and 2400Hz combination in the IMD test (b).

The second test also shows a good result which able to detect multiple frequency signals. The test is conducted with the same duration and sample rate with the previous test but with two different frequencies. There are three pairs of frequencies tested, i.e. 800Hz-4800Hz, 2400Hz-4800Hz, and 800Hz-2400Hz. One of the results is shown in Figure 46 (b). From the figure, we can see that inevitable harmonics and ripple occur in frequency greater than 4.1 kHz. These harmonics and ripple come from the sampling theorem. The higher we oversample the signal on a frequency, the better we obtain the signal. Hence, the signal in low frequency tends to be smoother than in the high frequency. But since the frequency greater than 4.1 kHz is not our interest, these harmonics and ripple are negligible.

V.6 Attenuation Characteristics Analysis

We have explained that there are two methods for measuring the damping ratio, i.e. the logarithmic decrement and the envelope fitting. In this research, we are implementing the damping ratio calculator tool with envelope fitting.

In the longitudinal stress wave technique, we use the microphone as a sensor. The microphone may receive another signal (noise) i.e. background noise in the room or the friction of wood and the support or base if we tap the wood too hard. Wood also an anisotropic material that has not only one natural frequency. It is not surprising if we have a signal with multiple frequency components in the recorded signal. This fact leads to several problems in this research. Figure 47 shows the spectrum of Acacia wood sampled with 44100 kHz sampling rate and it has a peak frequency of 1398 Hz. We can see that there is some unwanted frequency component which we don't know it is noise or it is the natural frequency of wood. Typically, what we considered as the fundamental natural frequency is the frequency component with the highest magnitude.

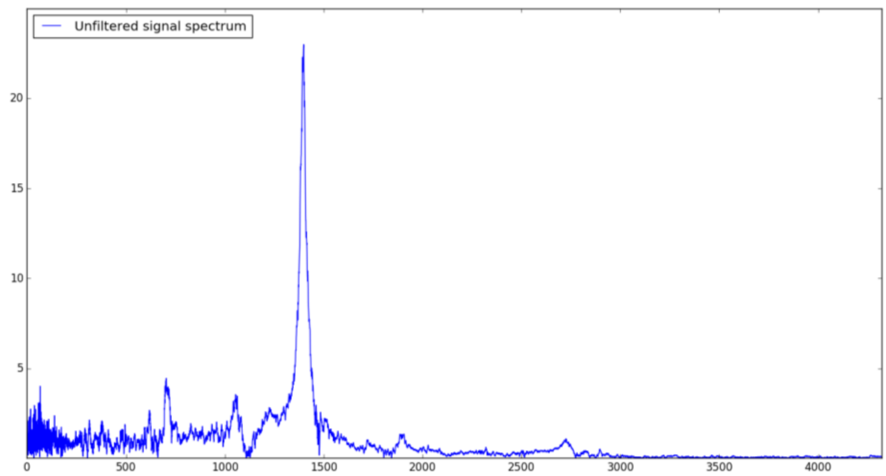


Figure 47. The example of tap signals spectrum of wood.

If we are curious with the signal spectrum of wood, we can observe the signal and observe a unique phenomenon Figure 48. The peak frequency of the longitudinal stress wave of wood is not a pure frequency, it is rippling, and it is likely some frequencies that very close each other. This phenomenon is reproducible Figure 49 even we can have this kind of phenomenon in many kinds of wood we have studied.

While for predicting the MOE, this phenomenon is can be neglected because we only considered the fundamental natural frequency and error in tens or hundreds of Hz is insignificant. For predicting the damping ratio from longitudinal stress wave, it is significant to consider this problem. It is related to how wide we set the cutoff frequency.

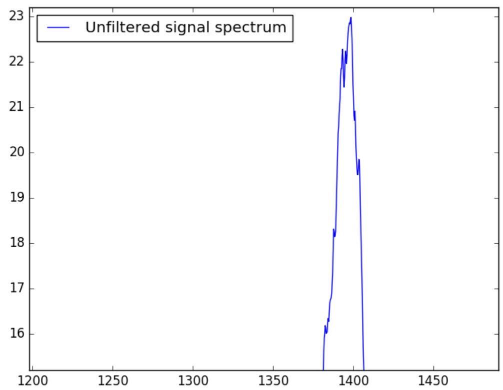
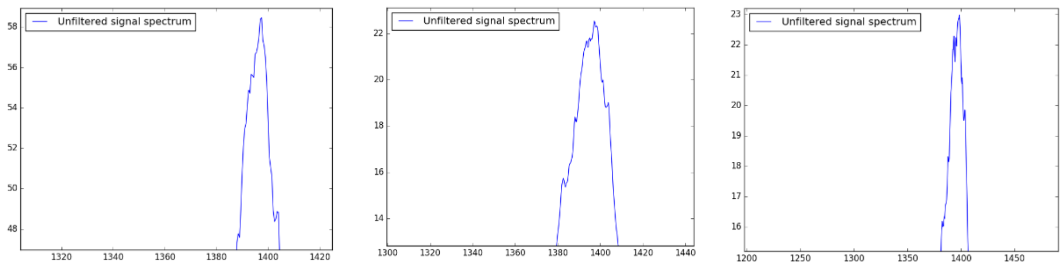


Figure 48. Zoom view of the example of tap signal spectrum of wood in Figure 47.



(a) **(b)** **(c)**

Figure 49. Ripple in the region near peak frequency of longitudinal stress wave for same wood in first, second and third measurement.

The first problem when measuring the damping ratio of the longitudinal stress wave is how to separate the fundamental natural frequency with another frequency component. For this problem, we used Butterworth band pass filter to tackle this problem. The Figure 50 shows the characteristic of Butterworth band pass filter for cutoff frequency 800 and 4100 Hz in various orders. We can see that the higher the order of the filter, the better "brick wall" created, but the higher the order, the higher probability the filter has pass band ripple. As we can see, we have pass band ripple for a 9th order filter for this cutoff frequency.

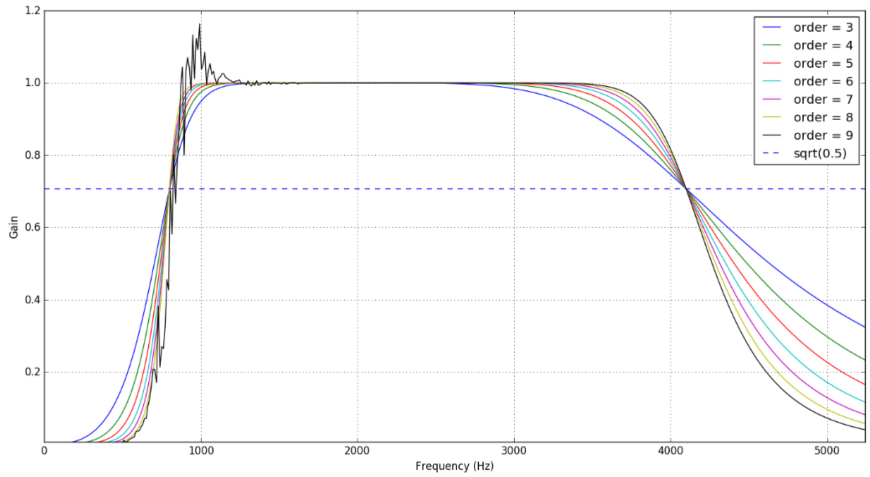
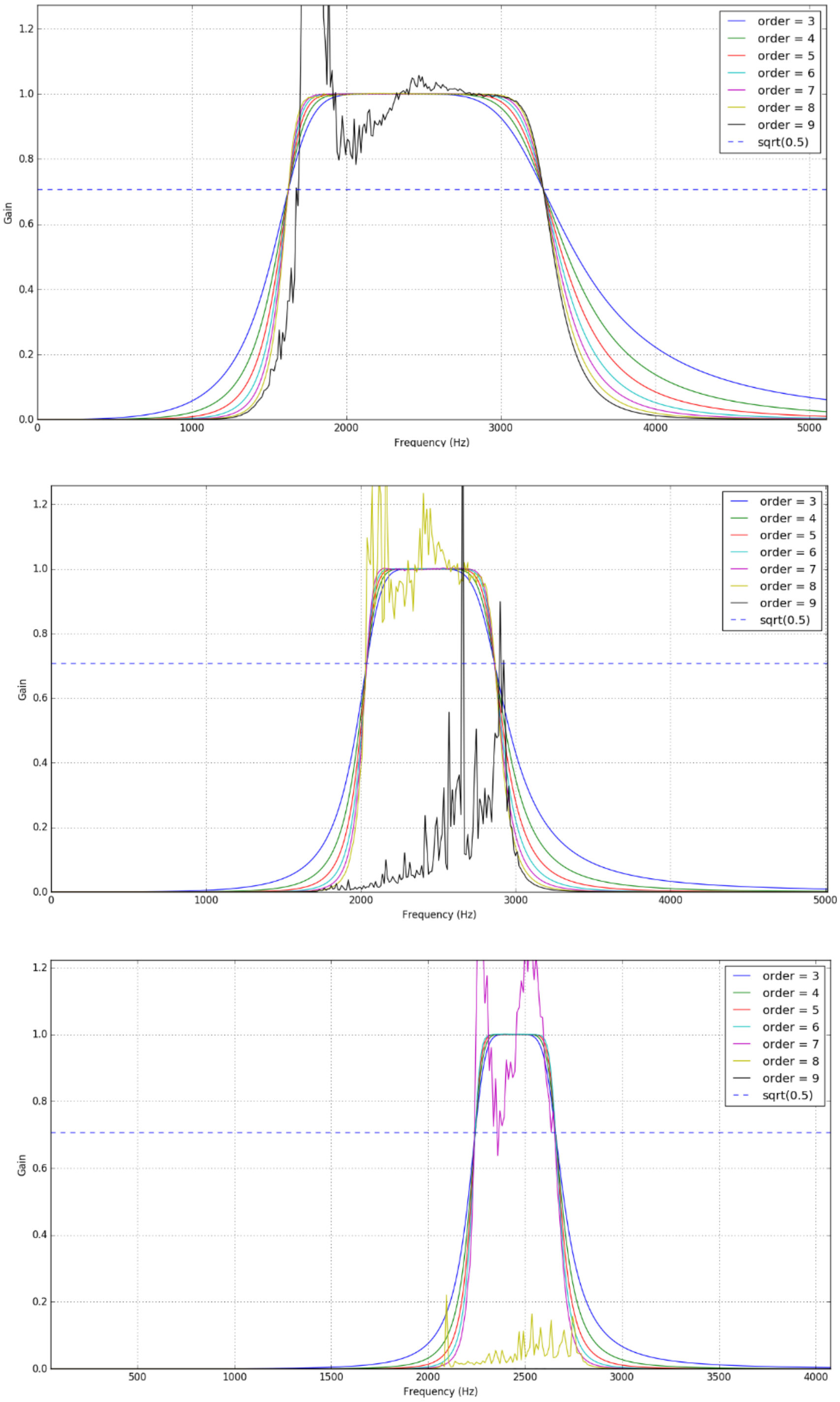


Figure 50. The Butterworth band-pass filter with cut-off frequency of 800-4100 Hz in various frequencies.

In Figure 50, the pass band ripple occurs in the 9^{th} order filter. The other properties of Butterworth filter are the smaller the frequency range, the higher probability smaller order filter has a pass band ripple. The pass band gain of the filter with smaller frequency range is also decreased in the relatively high order. We can see in Figure 51, in (a) the 9^{th} order filter has pass band ripple; in (b) the 8^{th} order and 9^{th} order filter has pass band ripple, but 9^{th} order filter has pass band gain lower than 1; in (c) the 9^{th} order filter has zero pass band gain, 8^{th} order filter has pass band gain lower than 1 and 7^{th} order filter has the pass band ripple.

From the observations of signal ripple near the peak frequency of longitudinal stress wave, and the properties of Butterworth filter, we set the default filter order for the command line tool is 3. So, we can handle an inevitable ripple near the peak frequency of longitudinal stress wave by including the ripple to envelope fitting calculation and keep the pass band gain flat. Throughout the test, we used 3^{rd} Butterworth filter to filter the longitudinal stress wave signal.



(a)

(b)

(c)

Figure 51. Pass band ripple in various cutoff frequency. Between 1625-3275 Hz (a). Between 2037-2863 Hz (b). Between 2243-2657 (c).

To avoid distortion of filtered signal, we study the effect of changing pass band frequency range to the result of filtered signal. For instance, we have a longitudinal stress wave signal in Figure 47. It has a peak frequency of 1398 Hz. From the graphic in the figure, we can safely say that the frequency component higher than 2000 Hz and lower than 1000 Hz is not a fundamental natural frequency component and we can set 1000 Hz and 2000 Hz as lower cut off and higher cut-off frequency respectively, but what is the optimal cut-off frequency for this kind of signal.

We compared several cut-off frequencies, i.e. peak signal ±400Hz, peak signal ± 200, peak signal ±100. The filtered signal and spectrum is shown in Figure 52. We have a tradeoff if we lower the pass band width, we distort the signal but we have a purer signal than the higher pass band width. Figure 52 (a) shows the filtered signal with a cut-off frequency equal to peak signal ±100 Hz. In this condition, we have a purer signal as shown in spectrum graph, but we have a distorted signal, we lost the peak impulse of the unfiltered signal. In (b), with a cut-off frequency equal to peak signal ±200 Hz, we can see the filtered signal is less pure than (a) but we have the impulse of the unfiltered signal. In (c), with a cut-off frequency equal to peak signal ±400 Hz, we have better signal in time domain, we get the impulse of the unfiltered signal, but have less pure frequency component. This condition is also applicable to another type of wood; we take Acacia wood just for an example.

From the observation, we set the default cut-off frequency of the bandpass filter is equal to peak signal ±200 Hz. So, we have a filtered signal in which contain as pure as possible the fundamental resonance frequency, and not distorting the original.

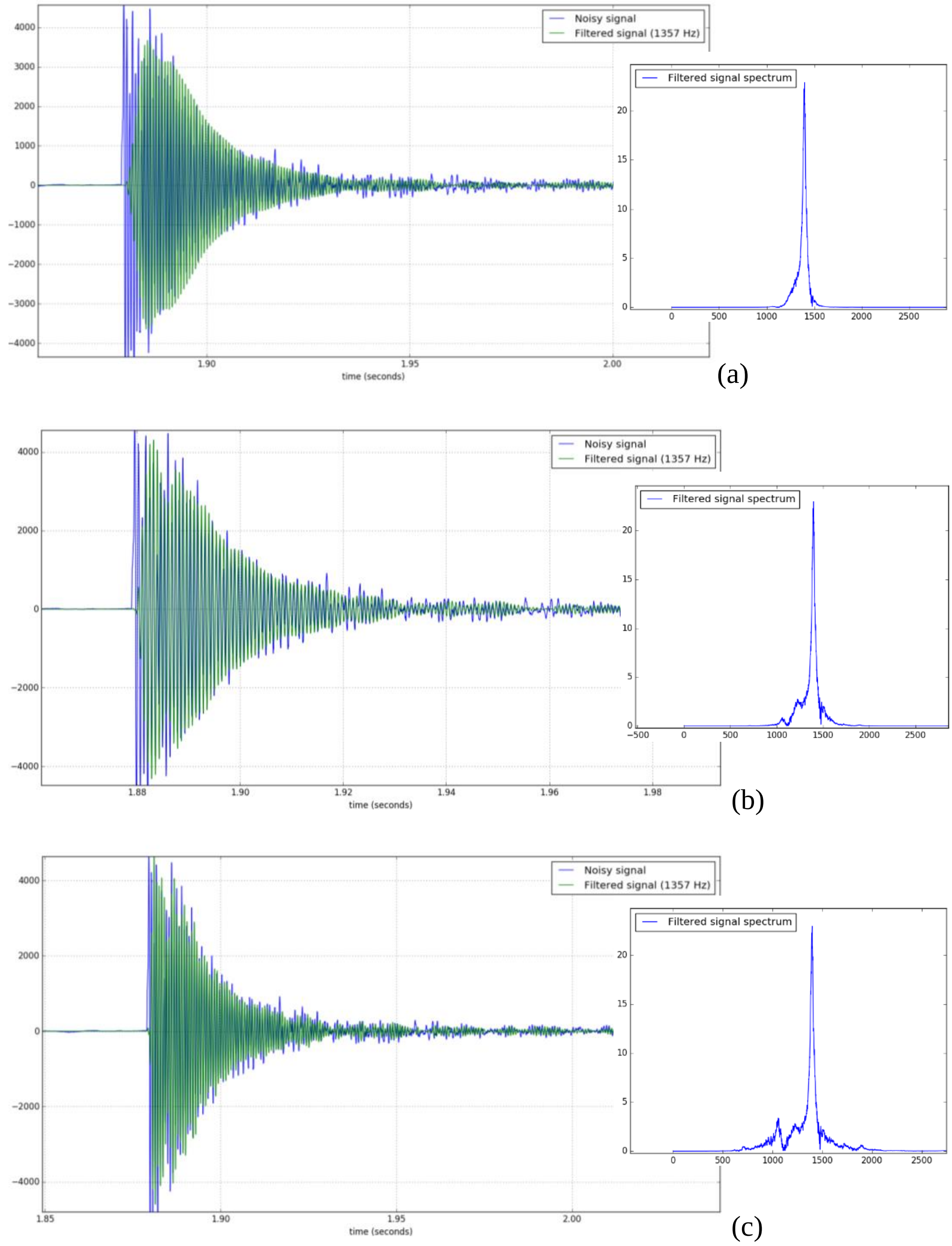
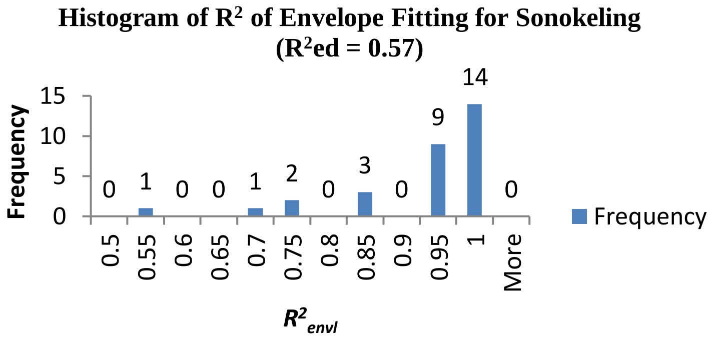
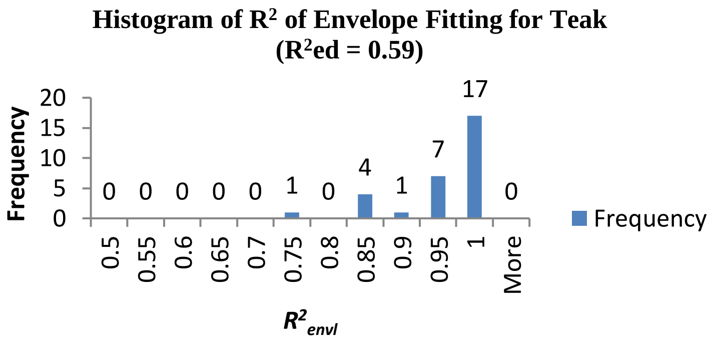


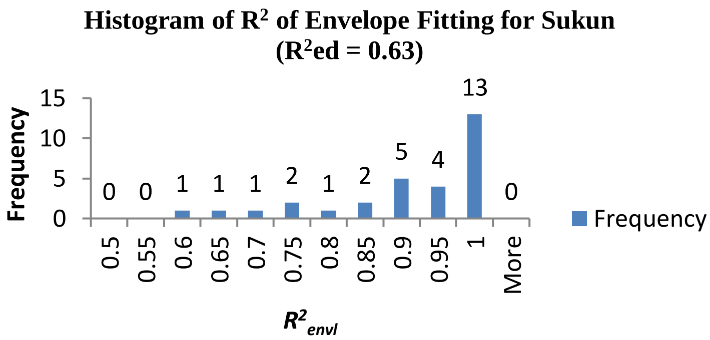
Figure 52. Filtered longitudinal stress wave signal with cut off frequency (a) peak signal ±100Hz, (b) peak signal ± 200 (c) peak signal ± 400.


Histogram of R² of Envelope Fitting for Sonokeling
(R²ed = 0.57)
Frequency
15
10
5
0
0 1 0 0 1 2 0 3 0 9 14 0
0.5 0.55 0.6 0.65 0.7 0.75 0.8 0.85 0.9 0.95 1 More
R²envl
Frequency

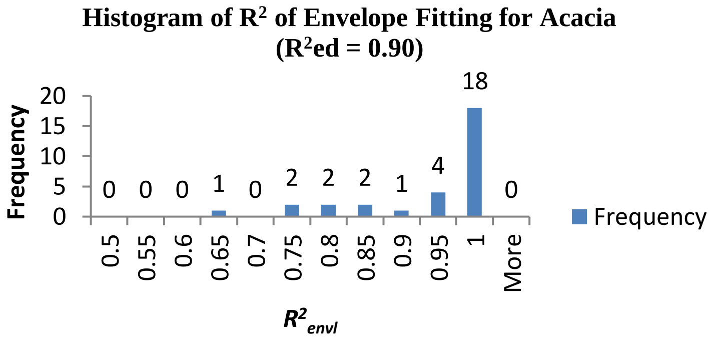
(a)


Histogram of R² of Envelope Fitting for Teak
(R²ed = 0.59)
Frequency
20
15
10
5
0
0 0 0 0 0 1 0 4 1 7 17 0
0.5 0.55 0.6 0.65 0.7 0.75 0.8 0.85 0.9 0.95 1 More
R²envl
Frequency

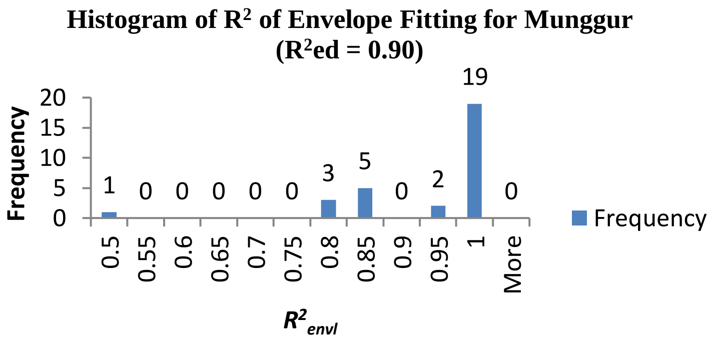
(b)


Histogram of R² of Envelope Fitting for Sukun
(R²ed = 0.63)
Frequency
15
10
5
0
0 0 1 1 1 2 1 2 5 4 13 0
0.5 0.55 0.6 0.65 0.7 0.75 0.8 0.85 0.9 0.95 1 More
R²envl
Frequency

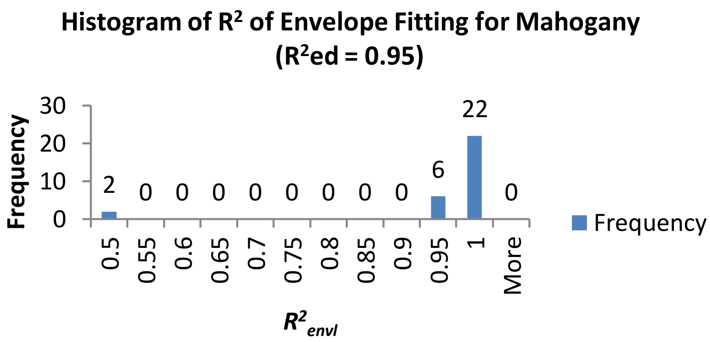
(c)



(d)



(e)



(f)

Figure 53. Histograms of R^2 of various wood, i.e. Sonokeling (a), Teak (b), Sukun (c), Acacia (d), Munggur (e), and Mahogany (f).

Figure 53 shows the histogram (distribution) of the envelope fitting R^2 of exactly same wood as researched by [7], [8]. We called this coefficient of determination as R^2_{envl}. In this section, we will compare this distribution with the R^2 of the MOE_d vs MOE_s correlation (called R^2_{ed} see Table 1). At glance, if we compare the R^2_{ed} in Table 6, the wood with low R^2_{ed} tend to have R^2_{envl} distribute exponentially. In contrast, the wood with high R^2_{ed} i.e. Acacia, Munggur, Mahogany tend to have R^2_{envl} around 0.95. A complete Raw data can be seen in Appendix 4.C.

Table 6. The pattern of R^2_{ed} and R^2_{envl}.

Wood Type	R^2_{ed}	**Percentage R^2_{envl} > 0.95**
Sonokeling	0.57	46.67 %
Teak	0.59	56.67 %
Sukun	0.63	43.33 %
Acacia	0.90	60.00 %
Munggur	0.90	63.33 %
Mahogany	0.95	73.33 %

The Table 6 shows clearer pattern based on our calculation. Woods with higher R^2_{ed} have the higher percentage of R^2_{envl} greater than 0.95. It means that the woods with higher R^2_{ed} have a better data. The data here is their recorded longitudinal stress wave. It's proved that the R^2 of the envelope fitting, R^2_{envl}, can be used to measure how well the recorded signal is representing the longitudinal stress wave as mentioned in the hypothesis. Besides that, it is also a supporting fact that the procedure of measuring the damping ratio from recorded longitudinal stress wave signal is correct because most calculation shows that it has R^2_{envl} greater than 0.95. Meaning, all the exponential decay follows the nature of damped vibration as discussed in section III.2.

V.7 How to Use the Instrument Correctly

V.7.a Generating a unit impulse input

A simple mechanism to generate a unit impulse input is using an impactor manually by hand. For a manual initiation, the following factor must be considered:

1) To get a homogenous wave propagation, it is best if we tap to the center of wood surface.
2) The size of impactor face is arbitrary as long as the surface is flat. Some researcher prefers a small impactor face to be able to release the impactor immediately after tapping the wood.

Another method is to create a tool to generate and control the quality of impulse. Either way, there is a universal known suggestion to do this, that is avoid clipping in the recording step. It is can be done by avoiding a very strong tap and put the microphone in the near field area to get a direct sound.

There are some ideas which not performed in this study, which are using a rebound hammer to generate an impulse and creating a hammer with a timer. First, using a rebound hammer or a hammer which automatically rebound after hit a surface to generate an impulse is a good idea if we can create a rebound time as small as possible.

The second idea is we use a hammer which has a timer to record the time from it rest condition to the time where it has highest velocity and becoming rest again. We can explore further for the maximum time for tapping if we want to use this method. Either way, the proposed enhanced longitudinal stress wave provides a set of tool to validate the longitudinal stress wave so that we can assess our recorded signal and do some fixes and repeat if necessary.

V.7.b Reverberation of Room

Every instrument is useless if it's not accurate. Instrument which we have developed is a tool to measure natural frequency of longitudinal stress wave (LSW) of wood. It is useful because if we can measure the natural frequency (f_0) of LSW, we can calculate the modulus of elasticity (MOE).

Because we rely on f_0 to calculate MOE, a careful attention must be paid to longitudinal stress wave measurement. One of the contributions of this research is to assess the recorded longitudinal stress wave of wood whether it is acceptable or contain noises. To do that we extract the envelope of the recorded signal and doing an exponential fitting to it. For a good recorded longitudinal stress wave, we expect to have an exponential attenuation.

However, the instrument always assume that input signal is coming from longitudinal stress wave of wood. We need to pay careful attention to the recording process if we want to guarantee that the input signal is coming from longitudinal stress wave. An interesting problem happen when we record or measure the longitudinal stress wave in an enclosed room or an acoustically reflective environment. In that situation, when a reverberation occurs, we will end up having a dirty signal (signal which not pure from wood).

Reverberation is the collection of reflected sounds from the surfaces in an enclosure like an auditorium or classroom. Basically, the reverberation is there for a reason. It is a desirable property of rooms to the extent that it helps to overcome the inverse square law drop off sound intensity in the enclosure. However, in our case, a reverberation may disturb our measurement.

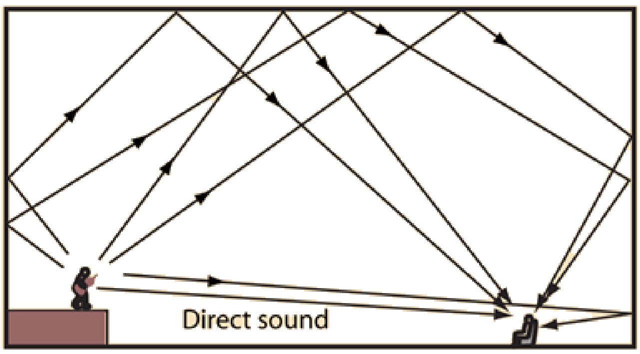


Figure 54. Reverberation in room [23].

The sketch in Figure 54 shows the sound received by a single listener as a function of time because of a sharp sound pulse some distance away. The direct sound received is followed by distinctly reflected sounds and then a collection of many reflected sounds which blend and overlap into what is called reverberation. There are two commonly used parameter to assess reverberation property of a room i.e. Early first-reflected sound time and Reverberation Time (RT_{60}). The early first-reflected sound time is the time needed for the first listener receive the first sound reflection, and the RT_{60} is he time to drop 60dB below original level.

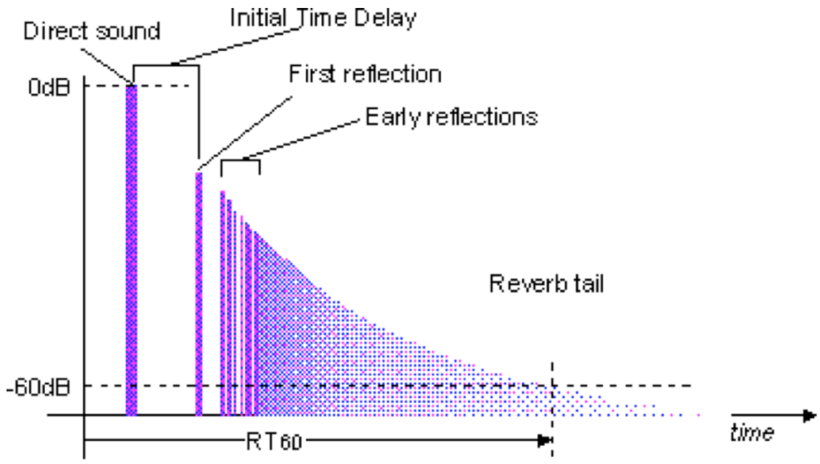


Figure 55. Sound receive by a listener in a reverberating room.

Basically, if reverberation exists in a room, after initiated a sound will live longer in a room. Whether from a sound impulse or a steady sound which ceases, the reverberant sound in a room decays in an approximately exponential fashion which similar to longitudinal stress wave initiated by an impulse response. If we conduct our measurement in a reflective room, we can have a recorded signal which contains a large portion of room resonant instead of having a pure longitudinal stress wave of wood.

From our observation, the typical duration of longitudinal stress wave from initial magnitude until decay to zero is 100ms (average). In a very reflective room we can have RT_{60} higher than 500 ms, so that it is easy to spot error from reverberation effect if we know the RT60 parameter of the room where the measurement taking place. The duration of the recorded LSW will increase if we record in a reflective room.

The problem is what is the RT_{60} for a typical room? Since the room will vary in size and materials which live on it, it will be hard to create a standard. Basically, an anechoic chamber is the best place to measure the longitudinal stress wave since we won't have any reflection (or only small) and we can control the background noise. But, since my instrument aims to be portable and mobile, a measurement in the unideal situation must be considered.

Basically, we can easily spot the reverberation effect since the duration of the longitudinal stress wave will be longer (a sloping exponential). If not, we will have a longitudinal stress wave with duration around 100 ms.

We can also analyze the effects of reverberating room in frequency domain. Originally, our setup in a diagram block would look like the Figure 56. The output signal is the convolution of the input signal (impulse response of wood) and the system (i.e. the reverberating room). In the frequency domain, our output spectrum is the multiplication of our input spectrum and the room's Transfer Function.

Figure 56. Block diagram of longitudinal stress wave measurement.

In our case, we expect to have a signal which only comes from Wood's impulse response. In a condition where reverberation is significantly involved, what we received in microphone would be different. In some frequencies, the multiplication result might constructive, and others frequencies might destructive. We don't know the exact result if we don't have an impulse response (or transfer function) of the Room.

The physical interpretation is the resulted reflection wave would interfere with original sound and would result in a constructive or destructive interference. And since we have multiple frequency components, we would have a different result in each frequency.

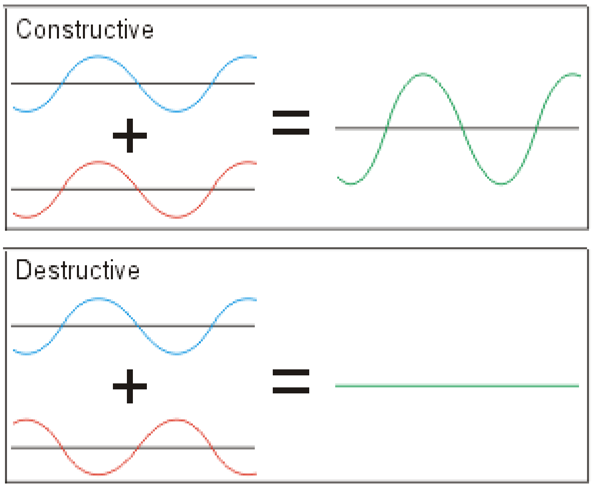


Figure 57. Constructive and destructive wave interferences.

So, conclude the following solution,

1) Create a warning if we have a longitudinal stress wave duration longer than 100ms.
2) In case we have a significant reverberation, we need to measure or characterize the Room's impulse response. We can create an impulse by popping a balloon in the room and record the response with an omnidirectional microphone. So, when we measure the longitudinal stress wave of wood inside the room, we can divide (or deconvoluting) the output signal with the impulse response of the room.

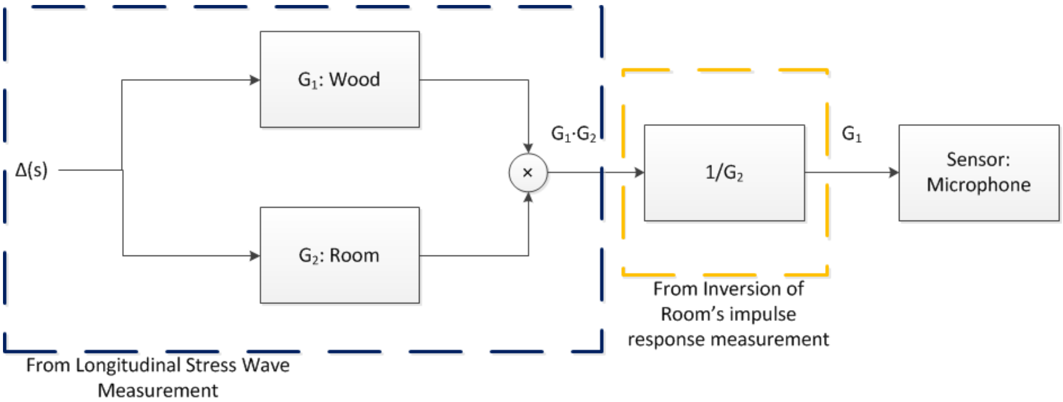


Figure 58. Room reverberation compensation setup.

CHAPTER VI
CONCLUSION AND RECOMMENDATION

VI.1 Conclusion

The conclusions of this research can be summarized as follows,

1. A prototype of an instrument which measure the fundamental natural frequency of longitudinal stress wave of wood is well developed. It has Raspberry Pi 2 Model B processor, a USB soundcard, powered with 1200 mAh battery, and a microphone with frequency range of 100–5000 Hz and -64 ± 3dB. We also recheck the frequency range of microphone in the frequency range of wood longitudinal stress wave in 800–4100 Hz. A measurement in soundproof room shows that the microphone has SNR > 0 dB in the expected frequency range.
2. From 180 recorded longitudinal stress wave of wood of 6 kinds of wood. There is a pattern where the R^2 of correlation between MOE_s and MOE_d is proportional with percentage of distribution of R^2_{envl} > 0.95. This result is used to perform longitudinal stress wave validation in instrument.
3. We successfully implement the validation step of longitudinal stress wave based on the result on 2).

VI.2 Recommendation

Per the research that has been done, several recommendations which can be proposed for further research are as follows,

- In this research, we are not exploring the possibility of using the damping factor, *h*, that has been calculated using our procedure to define a refined model of measuring the MOE. A depth study about this area may enrich our knowledge of nondestructive evaluation of wood.
- We test the implemented design's microphone with a flat response speaker. A further research implemented proposed design's microphone with tuning fork or a standardized calibrator will be appreciated.

REFERENCES

[1] Wirendro Sumargo, S. G. Nanggara, F. A. Nainggolan, and I. Apriani, *Potret Keadaan Hutan Indonesia Periode Tahun 2000-2009*, Edisi Pert. Forest Watch Indonesia, 2011.

[2] World Richest Countries, "Top Exported Timber Countries 2015," 2016. [Online]. Available: http://www.worldsrichestcountries.com/top-exported-timber-countries.html. [Accessed: 16-Oct-2016].

[3] World Richest Countries, "Top Exported Timber Countries 2014," 2016. [Online]. Available: https://web.archive.org/web/20160412183529/http://www.worldsrichestcountries.com/top-exported-timber-countries.html. [Accessed: 12-Apr-2016].

[4] Badan Standarisasi Nasional Indonesia, *SNI 7973-2013 Spesifikasi Desain untuk Konstruksi Kayu*, 3rd Rev. 2013.

[5] T. Stoebe, "Young's Modulus," *Materials Science and Engineering Education University of Washington*, 2000. [Online]. Available: http://depts.washington.edu/matseed/mse_resources/Webpage/Biomaterials/young's_modulus.htm. [Accessed: 16-Oct-2016].

[6] R. J. Ross, Ed., *Nondestructive Evaluation of Wood*, 2nd Ed. Forest Product Laboratory U.S. Department of Agriculture, 2015.

[7] F. Feliana, "Studi Empiris Nilai Modulus Elastisitas Kayu Menggunakan Metode 'Stress Wave Velocity,'" Universitas Gadjah Mada, 2014.

[8] Ayutyastuti, "Studi Kelayakan Metode Nondestructive Test Berbasis Akustik untuk Memprediksi Nilai Modulus Elastisitas Kayu," Universitas Gadjah Mada, 2015.

[9] N. Sobue, "Measurement of Young's Modulus by the Transient Longitudinal Vibration of Wooden Beams Using a Fast Fourier Transformation Spectrum Analyze." 日本木材学会, 1986.

[10] M. Ciornei, E. Diaconescu, and M. Glovnea, "Experimental Investigations of Wood Damping and Elastic Modulus," *Doct-Us*, vol. 31, pp. 2–7, 2009.

[11] physicsnet.co.uk, "Forced Vibrations & Resonance," 2017. [Online]. Available: http://physicsnet.co.uk/a-level-physics-as-a2/further-

mechanics/forced-vibrations-resonance/. [Accessed: 08-Apr-2017].

[12] J. Griffin, “Log Decrement,” *Carnegie Mellon Univ. Dyn. Syst. Control.*, p. 1, 2006.

[13] E. H. Currie and D. Van Ess, “PSoC3 / 5 Reference Book,” 2010.

[14] M. A. Meyers, *Dynamic Behavior of Materials*. New York: John Wiley & Sons Inc., 1994.

[15] P. Elsea, “Microphones,” 1996. [Online]. Available: http://artsites.ucsc.edu/ems/music/tech_background/te-20/teces_20.html. [Accessed: 21-Feb-2017].

[16] T. K. Moon, “Sampling Theorem,” pp. 1–4.

[17] J. Vanderplas, “Understanding the FFT Algorithm,” *Pythonic Perambulations*, 2013. [Online]. Available: https://jakevdp.github.io/blog/2013/08/28/understanding-the-fft/. [Accessed: 28-Dec-2016].

[18] J. W. Cooley and J. W. Tukey, “An Algorithm for Machine Calculation of Complex Fourier Series,” *Math. Comput.*, vol. 19, no. 90, p. 297-, 1965.

[19] AspenCore Inc, “Butterworth Filter Design with a Low Pass Butterworth,” 2016. [Online]. Available: http://www.electronics-tutorials.ws/filter/filter_8.html. [Accessed: 24-Feb-2017].

[20] L. Mol, L. A. Rocha, E. Cretu, and R. F. Wolffenbuttel, “Squeezed film damping measurements on a parallel-plate MEMS in the free molecule regime,” *J. Micromechanics Microengineering*, vol. 19, no. 7, p. 74021, Jul. 2009.

[21] “Signal-to-noise ratio,” *scholarpedia.org*.

[22] S. Pigeon, “Dual Frequency Generator (IMD Tests),” 2016. [Online]. Available: http://www.audiocheck.net/audiofrequencysignalgenerator_dual.php. [Accessed: 01-Jan-2016].

[23] HyperPhysics of Georgia State University, “Reverb.” [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/Acoustic/reverb.html. [Accessed: 03-Apr-2017].

[24] Dantheman, “Measurements on another Behringer monitor,” *PARTS Express*, 2007. [Online]. Available: http://techtalk.parts-express.com/forum/tech-talk-forum/37880-measurements-on-another-behringer-monitor. [Accessed: 10-Apr-2017].

APPENDIXES

APPENDIX.1 the Software Source Code

A. Project Structure

The project structure for the software is defined as follows

```
│    .gitignore
│    LICENSE
│    main.py
│    README.md
│    requirements.md
│
│
├───modules
│   │    __init__.py
│   │
│   ├───kayunde
│   │   │    calculation.py
│   │   │    fft.py
│   │   │    __init__.py
│   │
│   ├───kayurecord
│   │   │    recording.py
│   │   │    __init__.py
│   │
│   ├───tkinter_gui
│   │   │    utils.py
│   │   │    __init__.py
│
└───tests
    |___test_fft.py
```

We publicly publish the code under MIT License and the code can be downloaded here https://github.com/mappuji/ugm-kayu-nde.

B. Source: main.py

```
""" The main modules for the kayu Wood Nondestructive
Evaluation Software
"""

# pylint: disable=locally-disabled, W0201, W0401, W0613,
W0614, R0901, R0902, R0201, R0914, E1101

from tkinter import *
from tkinter import messagebox, filedialog
import tkinter as tk
import modules
from matplotlib.backends.backend_tkagg import
FigureCanvasTkAgg, NavigationToolbar2TkAgg
from matplotlib.figure import Figure
from numpy import arange, asarray, exp

# CONSTS
SCREEN_WIDTH = 480
SCREEN_HEIGHT = 320
TOP_FRAME_HEIGHT = SCREEN_HEIGHT - 40
BOTTOM_FRAME_HEIGHT = SCREEN_HEIGHT - 280
TOOLBAR_WIDTH = 40
```

```
MILLIS_IN_SECOND = 1000
EMPTY_STRING = ''
UPPER_THRESH_FREQ = 4100
LOWER_THRESH_FREQ = 800
MOE_UNIT = 'MPa'
FREQ_UNIT = 'kHz'
FREQ_UNIT_HZ = 'Hz'
FREQ_MAG_UNIT = 'dB'
LENGTH_UNIT = 'm'
DENSITY_UNIT = 'kg/m^3'
ENVELOPE_DATA_POINTS = 1764


class KayuApp(tk.Tk):
    """ Tkinter GUI class
    """

    def __init__(self, *args, **kwargs):
        tk.Tk.__init__(self, *args, **kwargs)
        container = tk.Frame(self)
        container.pack(side="top", fill="both", expand=True)
        container.grid_rowconfigure(0, weight=1)
        container.grid_columnconfigure(0, weight=1)
        startpage_frame = StartPage(container, self)
        startpage_frame.grid(row=0, column=0, sticky="nsew")
        self.title("Kayu: Open Wood NDE")
        self.frames = {}
        self.frames[StartPage] = startpage_frame
        self.show_frame(StartPage)

    def show_frame(self, cont):
        """ Show a frame
        """
        frame = self.frames[cont]
        frame.tkraise()

class StartPage(tk.Frame):
    """ Start page of the Tkinter GUI
    """

    def __init__(self, parent, controller):
        # inherit tk.Frame
        tk.Frame.__init__(self, parent)
        self.create_frame_structure()
        self.create_app_toolbar()
        self.create_app_graph()
        # variable for kayu
        self.kayu = {}
        self.kayu_weight = ''
        self.kayu_length = ''
        self.kayu_thick_x = ''
        self.kayu_thick_y = ''
        self.record_duration = ''
        self.kayu_name = ''
        self.kayu_naturfreq = ''
        # varibale for position button active position
        self.set_position = 0
        self.t_envel = []
        self.y_envel = []
        self.y_est = []

    def create_frame_structure(self):
        """ Create frame structure
```

```
        """
        self.frame_top = tk.Frame(
            self, width=SCREEN_WIDTH,
height=TOP_FRAME_HEIGHT)
        self.frame_bottom = tk.Frame(
            self, width=SCREEN_WIDTH,
height=BOTTOM_FRAME_HEIGHT)
        self.frame_toolbar = tk.Frame(
            self.frame_top, width=TOOLBAR_WIDTH,
height=TOP_FRAME_HEIGHT)
        self.frame_top.pack(side=TOP, expand=NO)
        self.frame_bottom.pack(side=BOTTOM, expand=NO,
anchor=SW)
        self.frame_toolbar.pack(side=RIGHT, expand=NO,
anchor=NE)

    def create_app_toolbar(self):
        """ Create app toolbar on the right side
        """
        self.button_settings = Button(self.frame_toolbar, \
            text="S", width=1, command=self.settings_window)
# settings
        self.button_record = Button(self.frame_toolbar, \
            text="R", width=1,
command=self.record_and_display) # records
        self.button_toggle = Button(self.frame_toolbar, \
            text="F", width=1, state="disabled",
command=self.display_freq_time)
            # toggles frequency and time domain
        self.button_pick = Button(self.frame_toolbar, \
            text="P", width=1, command=self.pick_naturfreq)
# picks frequency
        self.button_manu = Button(self.frame_toolbar, \
            text="V", width=1, command=self.verified_signal)
# manual inputs
        self.button_calc = Button(self.frame_toolbar, \
            text="C", width=1, command=self.calculated_moe)
# calulates MOE
        self.button_open = Button(self.frame_toolbar, \
            text="O", width=1, command=self.open_file)  #
open wavfile
        self.button_settings.pack(side=TOP, expand=NO,
anchor=NE)
        self.button_record.pack(side=TOP, expand=NO,
anchor=NE)
        self.button_toggle.pack(side=TOP, expand=NO,
anchor=NE)
        self.button_pick.pack(side=TOP, expand=NO,
anchor=NE)
        self.button_manu.pack(side=TOP, expand=NO,
anchor=NE)
        self.button_calc.pack(side=TOP, expand=NO,
anchor=NE)
        self.button_open.pack(side=TOP, expand=NO,
anchor=NE)

    def create_app_graph(self):
        """ Create app graph
        """
        self.graph_figure = Figure(figsize=(4.4, 2.8),
dpi=100)
        self.graph = FigureCanvasTkAgg(self.graph_figure,
master=self.frame_top)
```

```
        self.graph.get_tk_widget().pack(side=LEFT,
expand=NO, anchor=NW)
        # EMBED GRAPH TOOLBAR
        self.graph_toolbar =
NavigationToolbar2TkAgg(self.graph,
self.frame_bottom).pack(side=LEFT)
        # PLOT STARTING PLOT
        self.subplot = self.graph_figure.add_subplot(111)
        self.lines = self.subplot.plot([1, 2, 3], [1, 2, 3])
        self.change_xy_label('t(s)', 'magnitude')


    def record_and_display(self):
        """ Records a signal and display the recorded signal
        """
        duration = 5 if (self.record_duration == '') else
self.record_duration
        woodname = "placeholder" if (self.kayu_name == '')
else self.kayu_name
        wavfile = modules.kayurecord(woodname, duration)
        self.rate, self.signal = modules.kayuopen(wavfile)
        # plot a graph
        self.time_array = arange(len(self.signal))
        self.time_array = self.time_array / self.rate
        self.time_array = self.time_array * MILLIS_IN_SECOND
        self.lines = modules.plot_graph('modify',
self.lines, self.time_array, self.signal, \
            self.graph_figure)
        self.change_xy_label('t(ms)', 'magnitude')
        self.subplot.relim()
        self.subplot.autoscale_view()
        self.subplot.autoscale(enable=True, axis='both',
tight=None)
        self.graph.draw()

        # change button toggle state
        self.button_toggle['text'] = 'F'
        self.button_toggle['state'] = 'normal'

    def change_xy_label(self, xlabel, ylabel):
        """ Change xylabel of the plot
        """
        self.subplot.set_xlabel(xlabel)
        self.subplot.xaxis.set_label_coords(1.05, -0.05)
        self.subplot.set_ylabel(ylabel)
        self.subplot.yaxis.set_label_coords(-0.1, 0.5)

    def display_freq_time(self):
        """ Show spectrumm of the signal
        """
        if self.button_toggle['text'] == 'F':
            self.fft_ordinate, self.fft_abscissa =
modules.kayufft_db(self.signal, self.rate)
            self.lines = modules.plot_graph('modify',
self.lines,  \
                self.fft_abscissa, self.fft_ordinate,
self.graph_figure)
            self.change_xy_label('f(Hz)', 'magnitude (dB)')
            self.subplot.relim()
            self.subplot.autoscale_view()
            self.subplot.autoscale(enable=True, axis='both',
tight=None)
            self.graph.draw()
```

```
            self.button_toggle['text'] = 'T'

        elif self.button_toggle['text'] == 'T':
            # plot a graph
            self.time_array = arange(len(self.signal))
            self.time_array = self.time_array / self.rate
            self.time_array = self.time_array *
MILLIS_IN_SECOND
            self.lines = modules.plot_graph('modify',
self.lines, self.time_array, self.signal, \
                self.graph_figure)
            self.change_xy_label('t(ms)', 'magnitude')
            self.subplot.relim()
            self.subplot.autoscale(enable=True, axis='both',
tight=None)
            self.subplot.autoscale_view()
            self.graph.draw()
            self.button_toggle['text'] = 'F'

    def verified_signal(self):
        """ Verified recorded signal
        """
        if self.button_manu['text'] == 'V':
            messagebox.showinfo("Verified Signal", \
                "Through these steps you will verifying the
signal quality\n" \
                + "- means you'll see the filtered signal in
time\n" \
                + "+ means you'll see the filtered signal
spectrum\n" \
                + "* means you'll see the envelope\n" \
                + "/ means you'll see the fitting results")
            self.button_manu['text'] = '-'

        elif self.button_manu['text'] == '-':
            self.filtered_signal =
modules.kayu_damping_filter(self.signal,
self.kayu_naturfreq, \
                self.rate)
            # start
            self.lines = modules.plot_graph('modify',
self.lines, self.time_array, \
                self.filtered_signal, self.graph_figure)
            self.change_xy_label('t(ms)', 'magnitude')
            self.subplot.relim()
            self.subplot.autoscale(enable=True, axis='both',
tight=None)
            self.subplot.autoscale_view()
            self.graph.draw()
            # end
            self.button_manu['text'] = '+'

        elif self.button_manu['text'] == '+':
            # start
            self.fft_ordinate_fil, self.fft_abscissa_fil = \
                modules.kayufft_db(self.filtered_signal,
self.rate)
            self.lines = modules.plot_graph('modify',
self.lines,  \
                self.fft_abscissa_fil,
self.fft_ordinate_fil, self.graph_figure)
            self.change_xy_label('f(Hz)', 'magnitude (dB)')
            self.subplot.relim()
```

```
            self.subplot.autoscale_view()
            self.subplot.autoscale(enable=True, axis='both',
tight=None)
            self.graph.draw()
            # end
            self.button_manu['text'] = '*'

        elif self.button_manu['text'] == '*':
            # start
            start_point = self.filtered_signal.argmax()
            self.time_array_decay =
self.time_array[start_point:start_point +
ENVELOPE_DATA_POINTS]
            self.signal_decay =
self.filtered_signal[start_point:start_point +
ENVELOPE_DATA_POINTS]
            self.y_envel = [self.signal_decay[data] for data
\
                in range(len(self.signal_decay)-1) \
                if self.signal_decay[data-1] <
self.signal_decay[data] \
                and self.signal_decay[data+1] <
self.signal_decay[data]]
            self.t_envel = [(data + start_point) /
(self.rate / 1000) for data \
                in range(len(self.signal_decay)-1) \
                if self.signal_decay[data-1] <
self.signal_decay[data] \
                and self.signal_decay[data+1] <
self.signal_decay[data]]
            self.y_envel = asarray(self.y_envel)
            self.t_envel = asarray(self.t_envel)

            self.lines = modules.plot_double('modify',
self.lines, \
                self.time_array_decay, self.signal_decay, \
                self.t_envel, self.y_envel, \
                self.graph_figure)

            self.change_xy_label('t(ms)', 'magnitude')
            self.subplot.relim()
            self.subplot.autoscale(enable=True, axis='both',
tight=None)
            self.subplot.autoscale_view()
            self.graph.draw()

            # end
            self.button_manu['text'] = '/'

        elif self.button_manu['text'] == '/':
            (amplitude_est, tau_est) =
modules.fit_exponent(self.t_envel, self.y_envel)
            self.y_est = amplitude_est * (exp(-self.t_envel
/ tau_est))
            r_squared = modules.calc_r2(self.y_est,
self.y_envel)
            # start
            self.lines = modules.plot_double('modify',
self.lines, \
                self.t_envel, self.y_envel, \
                self.t_envel, self.y_est, \
                self.graph_figure)
            self.change_xy_label('t(ms)', 'magnitude')
```

```
            self.subplot.relim()
            self.subplot.autoscale(enable=True, axis='both',
tight=None)
            self.subplot.autoscale_view()
            self.graph.draw()
            messagebox.showinfo("R Square of the fitting",
str(r_squared))
            # stop
            self.button_manu['text'] = 'V'

    def toggle_a_button(self, button):
        """ Toggle a button
        """
        if button['state'] == 'normal':
            button['state'] = 'disabled'
        elif button['state'] == 'disabled':
            button['state'] = 'normal'

    def pick_naturfreq(self):
        """ Pick auto detect frequency natural or manual
        """
        naturfreq, naturfreq_mag = self.get_natural_freq()
        prompt = "Natural Frequency found at " +
str(naturfreq) + FREQ_UNIT_HZ + "\n" \
            + str(naturfreq_mag) + FREQ_MAG_UNIT + "\nDo you
want to use this frequency?"
        if messagebox.askyesno("Autodetect Natural
Frequency", \
             prompt):
            self.kayu_naturfreq = naturfreq
        else:
            messagebox.showinfo("Warning!", \
                "You need to input the desired natural
frequency manually!")

    def get_natural_freq(self):
        """ Get the natural frequency
        """
        argmax = modules.argmax_p(self.fft_ordinate,
self.rate)
        natural_freq = self.fft_abscissa[argmax]
        natural_freq_mag = modules.max_p(self.fft_ordinate,
self.rate)
        return natural_freq, natural_freq_mag

    def calculated_moe(self):
        """ Show the calculated MOE
        """

        weight = float(self.kayu_weight)
        length = float(self.kayu_length)
        thick_x = float(self.kayu_thick_x)
        thick_y = float(self.kayu_thick_y)
        naturfreq = float(self.kayu_naturfreq) / 1000

        self.kayu = modules.Kayu(weight, length, \
            thick_x, thick_y)
        self.kayu.set_freq(naturfreq)
        moe = self.kayu.get_moe()
        woodname_str = "The " + self.kayu_name + " wood
with\n"
        naturfreq_str = "Natural Freq = " +
str(self.kayu.get_freq()) + FREQ_UNIT + "\n"
```

```
        length_str = "Length = " + str(self.kayu.length) +
LENGTH_UNIT + "\n"
        density_str = "Density = " + str(self.kayu.density)
+ DENSITY_UNIT + "\n"
        moe_str = "Has MOE = " + str(moe) + MOE_UNIT +"\n"
        moe_sni = "Has SNI CLASS = " +
modules.sni_class(moe) +"\n"
        messagebox.showinfo("Results!", woodname_str +
naturfreq_str \
            + length_str + density_str + moe_str + moe_sni)

    def open_file(self):
        """ Open an audio file with file dialog
        """
        self.file_opt = options = {}
        options['defaultextension'] = '.wav'
        options['filetypes'] = [('all files', '.*'), ('Audio
File', '.wav')]
        options['initialdir'] = 'C:\\'
        options['initialfile'] = ''
        options['parent'] = self
        options['title'] = 'This is a title'
        self.wavfile = filedialog.askopenfilename(**options)
        self.rate, self.signal =
modules.kayuopen(self.wavfile)
        # plot a graph
        self.time_array = arange(len(self.signal))
        self.time_array = self.time_array / self.rate
        self.time_array = self.time_array * MILLIS_IN_SECOND
        self.lines = modules.plot_graph('modify',
self.lines, self.time_array, self.signal, \
            self.graph_figure)
        self.change_xy_label('t(ms)', 'magnitude')
        self.subplot.relim()
        self.subplot.autoscale_view()
        self.subplot.autoscale(enable=True, axis='both',
tight=None)
        self.graph.draw()

        # change button toggle state
        self.button_toggle['text'] = 'F'
        self.button_toggle['state'] = 'normal'

    def settings_window(self):
        """ Settings window
        """
        root = self
        title = "Setup Wood Data"
        settings_frame = Toplevel(root)
        # settings_frame.attributes('-fullscreen', True)
        settings_frame.wm_title(title)
        modules.win_center(settings_frame, 480, 320)
        settings_frame.lift(aboveThis=root)

        frame_input = Frame(settings_frame)
        frame_input.grid(column=0, row=0, sticky=(N))
        frame_keys = Frame(settings_frame)
        frame_keys.grid(column=0, row=1, sticky=(S))

        label_weight = Label(frame_input, text="Weight (kg)
")
        label_length = Label(frame_input, text="Length (m):
")
```

```
        label_crosssection_length = Label(frame_input,
text="CS Length (m): ")
        label_crosssection_width = Label(frame_input,
text="CS Width (m): ")
        label_duration = Label(frame_input, text="Duration
(s): ")
        label_woodname = Label(frame_input, text="Wood Name:
")
        label_naturfreq = Label(frame_input, text="Freq
Natural (Hz): ")
        label_weight.grid(column=0, row=0, sticky=(W))
        label_length.grid(column=0, row=1, sticky=(W))
        label_crosssection_length.grid(column=0, row=2,
sticky=(W))
        label_crosssection_width.grid(column=0, row=3,
sticky=(W))
        label_duration.grid(column=0, row=4, sticky=(W))
        label_woodname.grid(column=0, row=5, sticky=(W))
        label_naturfreq.grid(column=0, row=6, sticky=(W))

        weight_str = StringVar()
        entry_weight = Entry(frame_input,
textvariable=weight_str)
        entry_weight.grid(column=1, row=0)
        weight_str.set(self.kayu_weight)

        length_str = StringVar()
        entry_length = Entry(frame_input,
textvariable=length_str)
        entry_length.grid(column=1, row=1)
        length_str.set(self.kayu_length)

        crosssection_length_str = StringVar()
        entry_crosssection_length = Entry(frame_input,
textvariable=crosssection_length_str)
        entry_crosssection_length.grid(column=1, row=2)
        crosssection_length_str.set(self.kayu_thick_x)

        crosssection_width_str = StringVar()
        entry_crosssection_width = Entry(frame_input,
textvariable=crosssection_width_str)
        entry_crosssection_width.grid(column=1, row=3)
        crosssection_width_str.set(self.kayu_thick_y)

        duration_str = StringVar()
        entry_duration = Entry(frame_input,
textvariable=duration_str)
        entry_duration.grid(column=1, row=4)
        duration_str.set(self.record_duration)

        woodname_str = StringVar()
        entry_woodname = Entry(frame_input,
textvariable=woodname_str)
        entry_woodname.grid(column=1, row=5)
        woodname_str.set(self.kayu_name)

        naturfreq_str = StringVar()
        entry_naturfreq = Entry(frame_input,
textvariable=naturfreq_str)
        entry_naturfreq.grid(column=1, row=6)
        naturfreq_str.set(self.kayu_naturfreq)

        def set_weight():
```

```
            """ Function for button set weight
            """
            self.kayu_weight = weight_str.get()
            button_weight.config(state="disabled")
            button_length.config(state="normal")
            self.set_position += 1
            # print("succes")

        def set_length():
            """ Function for button set length
            """
            self.kayu_length = length_str.get()
            button_length.config(state="disabled")

button_crosssection_length.config(state="normal")
            self.set_position += 1
            # print("succes")

        def set_crosssection_length():
            """ Function for button set crossection_length
            """
            self.kayu_thick_x =
crosssection_length_str.get()

button_crosssection_length.config(state="disabled")
            button_crosssection_width.config(state="normal")
            self.set_position += 1
            # print("succes")

        def set_crosssection_width():
            """ Function for button set crossection_width
            """
            self.kayu_thick_y = crosssection_width_str.get()

button_crosssection_width.config(state="disabled")
            button_duration.config(state="normal")
            self.set_position += 1
            # print("succes")

        def set_duration():
            """ Function for button set record duration
            """
            self.record_duration = int(duration_str.get())
            button_duration.config(state="disabled")
            button_woodname.config(state="normal")
            self.set_position += 1
            # print("succes")

        def set_name():
            """ Function for button set name
            """
            self.kayu_name = str(woodname_str.get())
            button_woodname.config(state="disabled")
            button_naturfreq.config(state="normal")
            self.set_position += 1
            # print("succes")

        def set_naturfreq():
            """ Function for button set name
            """
            self.kayu_naturfreq = str(naturfreq_str.get())
            button_naturfreq.config(state="disabled")
            button_weight.config(state="normal")
```

```
            self.set_position = 0
            # print("succes")

        button_weight = Button(frame_input, text="Set",
command=set_weight)
        button_length = Button(frame_input, text="Set",
state=DISABLED, command=set_length)
        button_crosssection_length = Button(frame_input, \
            text="Set", state=DISABLED,
command=set_crosssection_length)
        button_crosssection_width = Button(frame_input, \
            text="Set", state=DISABLED,
command=set_crosssection_width)
        button_duration = Button(frame_input, text="Set",
state=DISABLED, command=set_duration)
        button_woodname = Button(frame_input, text="Set",
state=DISABLED, command=set_name)
        button_naturfreq = Button(frame_input, text="Set",
state=DISABLED, command=set_naturfreq)
        button_weight.grid(column=2, row=0)
        button_length.grid(column=2, row=1)
        button_crosssection_length.grid(column=2, row=2)
        button_crosssection_width.grid(column=2, row=3)
        button_duration.grid(column=2, row=4)
        button_woodname.grid(column=2, row=5)
        button_naturfreq.grid(column=2, row=6)

        def input_textbox(inputstring):
            """ Input textbox
            """
            if self.set_position == 0:
                modules.input_textbox(entry_weight,
inputstring)
            else:
                if self.set_position == 1:
                    modules.input_textbox(entry_length,
inputstring)
                else:
                    if self.set_position == 2:

modules.input_textbox(entry_crosssection_length,
inputstring)
                    else:
                        if self.set_position == 3:

modules.input_textbox(entry_crosssection_width, inputstring)
                        else:
                            if self.set_position == 4:

modules.input_textbox(entry_duration, inputstring)
                            else:
                                if self.set_position == 5:

modules.input_textbox(entry_woodname, inputstring)
                                else:

modules.input_textbox(entry_naturfreq, inputstring)

        def input_textbox7():
            """ input textbox
            """
            input_textbox('7')
        def input_textbox8():
```

```
            """ input textbox
            """
            input_textbox('8')
        def input_textbox9():
            """ input textbox
            """
            input_textbox('9')
        def input_textbox4():
            """ input textbox
            """
            input_textbox('4')
        def input_textbox5():
            """ input textbox
            """
            input_textbox('5')
        def input_textbox6():
            """ input textbox
            """
            input_textbox('6')
        def input_textbox1():
            """ input textbox
            """
            input_textbox('1')
        def input_textbox2():
            """ input textbox
            """
            input_textbox('2')
        def input_textbox3():
            """ input textbox
            """
            input_textbox('3')
        def input_textbox_dot():
            """ input textbox
            """
            input_textbox('.')
        def input_textbox0():
            """ input textbox
            """
            input_textbox('0')

        def input_textbox_del():
            """ delete a content of a textbox
            """
            if self.set_position == 0:
                modules.delete_textbox(entry_weight)
            else:
                if self.set_position == 1:
                    modules.delete_textbox(entry_length)
                else:
                    if self.set_position == 2:

modules.delete_textbox(entry_crosssection_length)
                    else:
                        if self.set_position == 3:

modules.delete_textbox(entry_crosssection_width)
                        else:
                            if self.set_position == 4:

modules.delete_textbox(entry_duration)
                            else:
                                if self.set_position == 5:
```

```
modules.delete_textbox(entry_woodname)
                                else:

modules.delete_textbox(entry_naturfreq)

        button7 = Button(frame_keys, text="7",
command=input_textbox7)
        button8 = Button(frame_keys, text="8",
command=input_textbox8)
        button9 = Button(frame_keys, text="9",
command=input_textbox9)
        button4 = Button(frame_keys, text="4",
command=input_textbox4)
        button5 = Button(frame_keys, text="5",
command=input_textbox5)
        button6 = Button(frame_keys, text="6",
command=input_textbox6)
        button1 = Button(frame_keys, text="1",
command=input_textbox1)
        button2 = Button(frame_keys, text="2",
command=input_textbox2)
        button3 = Button(frame_keys, text="3",
command=input_textbox3)
        button_dot = Button(frame_keys, text=".",
command=input_textbox_dot)
        button0 = Button(frame_keys, text="0",
command=input_textbox0)
        button_del = Button(frame_keys, text="Del",
command=input_textbox_del)
        button7.grid(column=1, row=0)
        button8.grid(column=2, row=0)
        button9.grid(column=3, row=0)
        button4.grid(column=1, row=1)
        button5.grid(column=2, row=1)
        button6.grid(column=3, row=1)
        button1.grid(column=1, row=2)
        button2.grid(column=2, row=2)
        button3.grid(column=3, row=2)
        button_dot.grid(column=1, row=3)
        button0.grid(column=2, row=3)
        button_del.grid(column=3, row=3)

        def quit_input():
            """ quit the settings window
            """
            self.set_position = 0
            # print(self.kayu_weight)
            # print(self.kayu_length)
            # print(self.kayu_thick_x)
            # print(self.kayu_thick_y)
            # print(self.record_duration)
            # print(self.kayu_name)
            # print(self.kayu_naturfreq)
            settings_frame.destroy()

        button_exit = Button(frame_input, text="Close",
command=quit_input)
        button_exit.grid(column=3, row=6)

    def underconstruction(self):
        """ Display underconstruction warning
        """
        messagebox.showinfo("Warning!", "This Functionality
```

```
        is under construction")

        # RUN THE TKINTER GUI
        APP = KayuApp()
        # APP.attributes('-fullscreen', True)
        modules.win_center(APP, SCREEN_WIDTH, SCREEN_HEIGHT)
APP.mainloop()
```

C. Source: calculation.py

```
""" Nondestructive evaluation of wood calculation
"""

# pylint: disable=locally-disabled, R0902, R0913

from math import pow as mathpow

class Kayu():
    """ Class for kayu - kayu is a term in Bahasa means wood
    Units:
            * All the units are in SI
            * kg, meter, kHz, MPa
    """

    def __init__(self, weight, length, thick_x, thick_y):
        self.weight = weight
        self.length = length
        self.thick_x = thick_x
        self.thick_y = thick_y
        self.volume = length * thick_x * thick_y
        self.density = weight / self.volume
        self.__freq = None
        self.__moe = None

    def set_freq(self, freq):
        """ Set the natural frequency of the wood
        """
        self.__freq = freq

    def get_freq(self):
        """ Get the natural frequency of the wood
             """
        return self.__freq

    def get_moe(self):
        """ Get the MOE of the wood
             """
        self.__moe = 4 * mathpow(self.__freq, 2) \
            * mathpow(self.length, 2) * self.density
#Formula of Dynamic MOE
        return self.__moe

class KayuPi(Kayu):
    """ The kayu class for recording via raspberry pi
    """
    def __init__(self, weight, length, thick_x, thick_y,
duration):
        Kayu.__init__(self, weight, length, thick_x,
thick_y)
 self.duration = duration
```

D. Source: fft.py

```
""" FFT calculation for wood nondestructive evaluation
"""

# pylint: disable=locally-disabled, W0612

from math import log
from pylab import matrix, lstsq
from numpy.fft import fft
from numpy import arange, ceil, log10, exp, mean
from numpy import sum as nsum
from scipy.signal import butter, lfilter


def kayufft_db(signal, rate):
    """ Perform FFT for wood's longitudinal stress wave
signal
        Inputs:
        Acoustic signal in time domain and the sampling rate
        Returns:
        Signal in ferquency domain with unit in dB
    """
    signal_length = len(signal)
    fft_result = fft(signal)
    uniq_points = int(ceil((signal_length + 1) / 2.0)) #only
takes one side of FFT result
    fft_result = fft_result[0:uniq_points]
    fft_result = abs(fft_result)
    fft_result = fft_result / float(signal_length)
    fft_result = fft_result**2
    if signal_length % 2 > 0:
        fft_result[1:len(fft_result)] =
fft_result[1:len(fft_result)] * 2
    else:
        fft_result[1:len(fft_result) - 1] =
fft_result[1:len(fft_result) - 1] * 2
    fft_abscissa = arange(0, uniq_points, 1.0) * (rate /
signal_length)
    fft_ordinate = 10 * log10(fft_result)
    return fft_ordinate, fft_abscissa

def argmax_p(fft_ordinate, rate):
    """ Find an argmax of fft result
    """
    upper_thresh_freq = 4100
    lower_thresh_freq = 800
    upper_thresh = int(len(fft_ordinate) / (rate / 2) *
upper_thresh_freq)
    lower_thresh = int(len(fft_ordinate) / (rate / 2) *
lower_thresh_freq)
    argmax =
fft_ordinate[lower_thresh:upper_thresh].argmax()
    argmax += lower_thresh
    return argmax

def max_p(fft_ordinate, rate):
    """ Find an argmax of fft result
    """
    upper_thresh_freq = 4100
    lower_thresh_freq = 800
    upper_thresh = int(len(fft_ordinate) / (rate / 2) *
upper_thresh_freq)
```

```
    lower_thresh = int(len(fft_ordinate) / (rate / 2) *
lower_thresh_freq)
    # return max(fft_ordinate)
    return max(fft_ordinate[lower_thresh:upper_thresh])

def butter_bandpass(lowcut, highcut, sampling_freq,
order=5):
    """ Butterworth with default order of 5
    """
    nyq = 0.5 * sampling_freq
    low = lowcut / nyq
    high = highcut / nyq
    numerator, denominator = butter(order, [low, high],
btype='band')
    return numerator, denominator

def butter_bandpass_filter(signal, lowcut, highcut,
sampling_freq, order=5):
    """ Butterworth filter with default order of 5
    """
    numerator, denominator = butter_bandpass(lowcut,
highcut, sampling_freq, order=order)
    y_array = lfilter(numerator, denominator, signal)
    return y_array

def kayu_damping_filter(signal, center_f, rate,
range_pm=200):
    """ Adaptive Butterworth filter for measuring wood's
modulus of elasticity
    """
    # center_f = argmax_p(p_db, rate)
    lowcut = center_f - range_pm
    highcut = center_f + range_pm
    filtered_signal = butter_bandpass_filter(signal, lowcut,
highcut, rate, order=3)
    return filtered_signal

def fit_exponent(t_list, y_list, y_steady=0):
    '''
    This function finds a
        t_list in sec
        y_list - measurements
        y_steady - the steady state value of y
    returns
        amplitude of exponent
        tau - the time constant
    '''
    b_list = [log(max(y-y_steady, 1e-6)) for y in y_list]
    b_list_transpose = matrix(b_list).T
    rows = [[1, t] for t in t_list]
    a_mat = matrix(rows)
    #w_params = (pinv(a_mat)*b_list_transpose)
    (w_params, residuals, rank, sing_vals) = lstsq(a_mat,
b_list_transpose)
    tau = -1.0/w_params[1, 0]
    amplitude = exp(w_params[0, 0])
    return (amplitude, tau)

def calc_r2(y_est, y_envel):
    """ Calculate the R2 of any fitting result
    """
    residuals = y_est - y_envel
    ss_res = nsum(residuals**2)
```

```
        ss_tot = nsum((y_envel - mean(y_envel))**2)
        r_squared = 1 - (ss_res / ss_tot)
        return r_squared

    def sni_class(moe):
        """ Classified SNI wood classification based on MOE
        """
        if moe >= 25000:
            return 'E25'
        if moe <= 5000:
            return 'E5'
        temp = moe // 1000
        sni = temp
        if moe % 1000 != 0:
            sni = sni + 1
return 'E' + str(sni)
```

E. Source: recording.py

```
""" Recording modules
"""

# pylint: disable=locally-disabled, C0330, W0612

import sys
import wave
import datetime as dt
import pyaudio
from scipy.io import wavfile as wav

# Global variables and const
# Configurations for the recording
# Sampling rate: 48000 Hz
RATE = 48000
CHUNK = 8192
FORMAT = pyaudio.paInt16
CHANNELS = 1
if sys.platform == 'darwin':
    CHANNELS = 1

# Functions, Classes & Procedures:
def time_now():
    """ Get current time
       """
    return dt.datetime.now().strftime("%Y%m%d%H%M%S")

def kayurecord_save(filename, frames, container):
    """ Save audio recording to wav file
    """
    wave_file = wave.open(filename, 'wb')
    wave_file.setnchannels(CHANNELS)

wave_file.setsampwidth(container.get_sample_size(FORMAT))
    wave_file.setframerate(RATE)
    wave_file.writeframes(b''.join(frames))
    wave_file.close()

def kayurecord(woodname, duration):
    """ Record audio and save to wav file
    """
    filename = time_now() + "_" + woodname + ".wav"
    container = pyaudio.PyAudio()
```

```
        stream = container.open(format=FORMAT,
                  channels=CHANNELS,
                  rate=RATE,
                  input=True,
                  frames_per_buffer=CHUNK)
        print("* start recording...")
        data = []
        frames = []
        for i in range(0, int(RATE / CHUNK * duration)):
            data = stream.read(CHUNK)
            frames.append(data)
        stream.stop_stream()
        stream.close()
        container.terminate()
        print("* done recording!")
        kayurecord_save(filename, frames, container)
        return filename

    def kayuopen(woodname):
        """ Open wav file
        """
        rate, data = wav.read(woodname)
        data = data / (2.**15)
        try:
            ch1 = data[:, 0]
        except IndexError:
            ch1 = data
return rate, ch1
```

F. Source: utils.py

```
""" Utilities module for tkinter_gui of nde software
"""

# pylint: disable=locally-disabled, R0913

def win_center(root_tk, x_pos, y_pos):
    """ Create tk windor to center position
    """
    # get screen width and height
    screen_width = root_tk.winfo_screenwidth() # width of
the screen
    screen_height = root_tk.winfo_screenheight() # height of
the screen

    # enable the code below if want to turn full size
    # root_width = root.winfo_screenwidth()
    # root_height = root.winfo_screenheight()
    root_width = x_pos
    root_height = y_pos

    # calculate x_pos and y_pos coordinates for the Tk root
window
    x_pos = (screen_width/2) - (root_width/2)
    y_pos = (screen_height/2) - (root_height/2)
    root_tk.geometry('%dx%d+%d+%d' % (root_width,
root_height, x_pos, y_pos))

def plot_graph(command, lines, x_axis, y_axis, figure):
    """ Plot graph using matplotlib backend
    """
    if command == 'create':
```

```
            subplot = figure.add_subplot(111)
            subplot.relim()
            subplot.autoscale_view()
            return subplot.plot(x_axis, y_axis, 'b')
        if command == 'modify':
            lines_list = lines[0]
            lines_list.remove()
            try:
                lines_list2 = lines[1]
                lines_list2.remove()
            except IndexError:
                pass
            subplot = figure.add_subplot(111)
            subplot.relim()
            subplot.autoscale_view()
            return subplot.plot(x_axis, y_axis, 'b')
        if command == 'destroy':
            lines_list = lines[0]
            lines_list.remove()
        return lines

    def plot_double(command, lines, x_axis, y_axis, x_axis2,
    y_axis2, figure):
        """ Plot graph using matplotlib backend
        """
        if command == 'create':
            subplot = figure.add_subplot(111)
            subplot.relim()
            subplot.autoscale_view()
            return subplot.plot(x_axis, y_axis, 'k-', x_axis2,
    y_axis2, 'r-')
        if command == 'modify':
            lines_list = lines[0]
            lines_list.remove()
            try:
                lines_list2 = lines[1]
                lines_list2.remove()
            except IndexError:
                pass
            subplot = figure.add_subplot(111)
            subplot.relim()
            subplot.autoscale_view()
            return subplot.plot(x_axis, y_axis, 'k-', x_axis2,
    y_axis2, 'r-')
        if command == 'destroy':
            lines_list = lines[0]
            lines_list.remove()
        return lines

    def input_textbox(keys, the_value):
        """ input a value textbox
        """
        keys.insert('end', the_value)

    def delete_textbox(keys):
        """ delete a value textbox
        """
keys.delete(0, 'end')
```

G. Source: test_fft.py

```
""" test fft modules
"""

# pylint: disable=locally-disabled, C0401, C0413

import sys
sys.path.append('..')
import modules

def test_sni():
    """ test the sni classification
    """
    assert modules.sni_class(26500) == 'E25'
    assert modules.sni_class(25000) == 'E25'
    assert modules.sni_class(24000) == 'E24'
    assert modules.sni_class(13000) == 'E13'
    assert modules.sni_class(13600) == 'E14'
    assert modules.sni_class(13500) == 'E14'
    assert modules.sni_class(13400) == 'E14'
    assert modules.sni_class(4000) == 'E5'
assert modules.sni_class(2500) == 'E5'
```

APPENDIX.2 Mathematical Derivations

A. Derivation of Damped Free Vibration Model

Damped free vibration can be modelled as motion of a spring that is a subject of a frictional force or a damping force. Generic form of this kind of damped vibration is

$$m\frac{d^2x}{dt^2}+c\frac{dx}{dt}+kx=0 \tag{A.1}$$

with mass, $m > 0$, damping constant, $c \geq 0$, and spring constant, $k > 0$. It is a second-order linear differential equation and its auxiliary equation is

$$ms^2+cs+k=0 \tag{A.2}$$

with the following roots

$$r_1,r_2=\frac{-c\pm\sqrt{c^2-4mk}}{2m} \tag{A.3}$$

There are three cases depending on the sign of the expression under the square root:

1) $c^2<4mk$ (this will be **underdamping**; c is small relative to m and k).
2) $c^2>4mk$ (this will be **overdamping**; c is large relative to m and k).
3) $c^2=4mk$ (this will be **critical damping**; c is just between over and underdamping).

Mathematically, the easiest case is overdamping because the roots are both real. However, since we are dealing with damped vibration we will only discuss about underdamping situation.

If $c^2<4mk$ then the term under the square root is negative and the characteristic roots are not real. For $c^2<4mk$ the damping constant b must be relatively small.

First we use the roots (A.2) to solve equation (A.1) let $\omega_d=\sqrt{|c^2-4mk|}/2m$ then we have characteristic roots $\frac{-c}{2m}\pm i\omega_d$ and leading to complex solutions $\exp\{(-c/2m+i\omega_d)t\}$ and $\exp\{(-c/2m-i\omega_d)t\}$.

The basic real solutions are $\exp(-ct/2m)+\cos(\omega_d t)$ and $\exp(-ct/2m)+\sin(\omega_d t)$. The general solution is found by taking linear combinations of the two basic solutions, that is:

$$x(t) = c_1 \exp(-ct/2m)\cdot\cos(\omega_d t) + c_2 \exp(-ct/2m)\cdot\sin(\omega_d t)$$

or

$$x(t) = \exp(-ct/2m)\cdot(c_1\cos(\omega_d t) + c_2\sin(\omega_d t)) = x_0\cdot\exp(-ht)\cdot\cos(\omega_d t - \phi) \qquad \text{(A.4)}$$

where h is damping factor satisfy $h = c/2m$. It is common to expressed $h = \zeta\omega_n$, where $\omega_n = \sqrt{k/m}$ and $\zeta = \sqrt{c^2/4km}$.

Let's analyze this physically. When $c = 0$ the $h = 0$ and the exponential part of (A.4) is contribute nothing to the whole equation and the resulting response is sinusoid. Damping is frictional force, so it generates heat and dissipates energy. When damping constant c is small we would expect the system to still oscillate, but with decreasing amplitude as its energy is converted to heat. Over time it should come to rest at equilibrium. This is exactly what we see in (A.4). The factor $\cos(\omega_d t - \phi)$ shows the oscillation. The exponential factor $\exp(-ht)$ has a negative exponent and therefore gives the decaying amplitude. As t becoming infinite, the exponential goes asymptotically to 0, so $x(t)$ also goes asymptotically to its equilibrium position $x = 0$.

We call ω_d the damped angular (or circular) frequency of the system. This is sometimes called a pseudo-frequency of $x(t)$. We need to be careful to call it a pseudo-frequency because $x(t)$ is not periodic and only periodic functions have a frequency. Nonetheless, $x(t)$ does oscillate, crossing $x = 0$ twice each pseudo-period.

APPENDIX.3 Referred Data

We referred to some experimental result from various research and literature. Some intensively examined data are presented here.

Feliana (2014) [7]

A. Static and Dynamic Evaluation of Teak Wood

Sample	Material Data					Dynamic Evaluation		Static Evaluation						
	h	b	Lo	W	ρ	f	MOE_d	F_{max}	F_1	w_1	F_2	w_2	MOE_s	MOR
	(m)	(m)	(m)	(kg)	(kg/m^3)	(kHz)	(MPa)	(kN)	(kN)	(mm)	(kN)	(mm)	(MPa)	(MPa)
Teak 1	0.120	0.060	1.260	7.980	880	1.3005	8210	17.51	1.80	3.96	6.83	14.01	5175	44
Teak 2	0.120	0.060	1.260	8.910	982	1.3449	9803	24.84	2.48	4.34	9.83	15.78	6653	62
Teak 3	0.120	0.060	1.260	8.005	882	1.5563	11801	20.08	2.05	3.39	7.93	12.10	6979	50
Teak 4	0.121	0.061	1.232	6.920	761	1.5499	11099	16.04	1.98	2.69	6.27	8.48	7237	38
Teak 5	0.121	0.061	1.231	7.290	802	1.5279	11353	25.59	2.60	3.00	10.13	9.96	10670	61
Teak 6	0.121	0.061	1.232	7.895	868	1.6330	14057	22.94	2.46	2.21	8.68	7.96	10147	54
Teak 7	0.121	0.060	1.230	7.245	811	1.7381	14833	25.59	2.58	3.22	10.01	10.26	10843	63
Teak 8	0.121	0.061	1.231	6.395	704	1.4754	9287	21.37	2.87	2.69	8.19	8.32	9222	51
Teak 9	0.121	0.060	1.232	6.730	752	1.6767	12843	36.42	3.71	3.74	13.89	12.26	12264	90
Teak 10	0.121	0.061	1.230	7.130	785	1.3178	8254	25.25	2.53	3.19	10.00	10.29	10272	61
Average ($\dot{X}$)						1.5121	11154	Average ($\dot{X}$)					8946	58
Minimum						1.3005	8210	Minimum					5175	38
Maximum						1.7381	14833	Maximum					12264	90

B. Static and Dynamic Evaluation of Acacia Wood

Sample	Material Data					Dynamic Evaluation		Static Evaluation						
	h	b	Lo	W	ρ	f	MOE_d	F_{max}	F_1	w_1	F_2	w_2	MOE_s	MOR
	(m)	(m)	(m)	(kg)	(kg/m^3)	(kHz)	(MPa)	(kN)	(kN)	(mm)	(kN)	(mm)	(MPa)	(MPa)
Acacia 1	0.0617	0.1205	1.233	8.675	946	1.4171	11556	23.60	2.41	1.95	9.42	9.60	9443	57
Acacia 2	0.0590	0.1210	1.233	8.305	943	1.3353	10230	31.30	3.21	4.95	12.51	16.72	8109	79
Acacia 3	0.0573	0.1205	1.233	8.380	984	1.4360	12343	20.10	2.06	2.91	8.10	11.37	7367	52
Acacia 4	0.0607	0.1205	1.233	8.885	985	1.4579	12734	26.70	2.71	3.40	10.59	12.30	9137	66
Acacia 5	0.0603	0.1205	1.232	8.280	925	1.4010	11022	24.16	2.42	3.60	9.64	12.70	8180	66
Acacia 6	0.0594	0.1214	1.234	8.525	958	1.4769	12728	24.17	2.47	3.93	9.65	13.48	7692	60
Acacia 7	0.0603	0.1202	1.232	7.150	801	1.4448	10148	20.42	2.08	2.67	8.14	10.31	8193	51
Acacia 8	0.0580	0.1255	1.232	8.840	986	1.4142	11969	20.03	2.01	2.89	7.99	10.62	7657	50
Acacia 9	0.0607	0.1199	1.232	8.235	918	1.3134	9619	22.17	2.24	3.47	8.85	13.12	7090	55
Acacia 10	0.0602	0.1201	1.233	8.765	983	1.4010	11736	20.53	2.12	2.64	8.26	10.17	8443	51
Average ($\dot{X}$)						1.410	11409	Average ($\dot{X}$)					8131	59
Minimum						1.313	9619	Minimum					7090	50
Maximum						1.477	12734	Maximum					9443	79

C. Static and Dynamic Evaluation of Mahogany Wood

Sample	Material Data					Dynamic Evaluation		Static Evaluation						
	h	**b**	**Lo**	**W**	**ρ**	**f**	**MOE_d**	**F_{max}**	**F_1**	**w_1**	**F_2**	**w_2**	**MOE_s**	**MOR**
	(m)	**(m)**	**(m)**	**(kg)**	**(kg/m^3)**	**(kHz)**	**(MPa)**	**(kN)**	**(kN)**	**(mm)**	**(kN)**	**(mm)**	**(MPa)**	**(MPa)**
Mahogany 1	0.060	0.117	1.233	5.540	637	1.4841	8538	9.63	0.99	1.86	3.82	6.73	6184	25
Mahogany 2	0.060	0.120	1.233	6.430	724	1.2391	6757	11.87	1.28	2.72	4.70	9.79	4998	25
Mahogany 3	0.058	0.120	1.233	5.985	695	1.2959	7098	13.07	1.33	3.54	5.16	11.94	4852	33
Mahogany 4	0.060	0.120	1.233	5.635	629	1.6155	9982	15.16	1.53	2.86	6.05	9.47	7058	38
Mahogany 5	0.059	0.121	1.233	6.210	713	1.2010	6256	5.20	0.53	1.51	2.04	5.96	3508	13
Mahogany 6	0.061	0.121	1.232	5.920	658	1.5016	9008	21.64	2.21	4.45	8.59	14.48	6562	53
Mahogany 7	0.058	0.120	1.232	5.570	646	1.6155	10240	23.88	2.45	4.12	9.53	13.82	7558	61
Mahogany 8	0.059	0.123	1.232	5.775	650	1.5893	9972	17.26	1.79	2.95	6.90	10.42	6923	43
Mahogany 9	0.058	0.120	1.230	5.955	702	1.2959	7136	11.29	1.33	2.86	4.49	9.51	4918	29
Mahogany 10	0.061	0.121	1.240	5.860	644	1.3923	7678	15.12	1.53	2.45	6.02	9.96	6127	37
Average ($\dot{X}$)						1.423	8266.527	Average ($\dot{X}$)					5869	36
Minimum						1.201	6256	Minimum					3508	13
Maximum						1.616	10240	Maximum					7558	61

D. Static and Dynamic Evaluation of Sonokeling Wood

Sample	Material Data					Dynamic Evaluation		Static Evaluation						
	h	b	Lo	W	ρ	f	MOE_d	F_{max}	F_1	w_1	F_2	w_2	MOE_s	MOR
	(m)	(m)	(m)	(kg)	(kg/m³)	(kHz)	(MPa)	(kN)	(kN)	(mm)	(kN)	(mm)	(MPa)	(MPa)
Sonokeling 1	0.058	0.120	1.230	7.105	830	1.1777	6966	25.05	2.53	3.55	9.99	12.92	8242	65
Sonokeling 2	0.060	0.120	1.231	7.855	886	1.6111	13944	18.45	1.89	2.68	7.36	9.75	8007	46
Sonokeling 3	0.060	0.120	1.231	7.450	841	1.6330	13587	24.56	2.51	3.73	9.77	12.74	8342	61
Sonokeling 4	0.060	0.121	1.231	7.330	820	1.6155	12975	25.71	2.62	4.21	10.25	13.91	8080	64
Sonokeling 5	0.059	0.122	1.230	6.820	776	1.1645	6365	21.16	2.16	3.55	8.43	12.64	7020	48
Sonokeling 6	0.057	0.119	1.231	7.630	914	1.5937	14067	19.42	1.97	3.07	7.76	10.49	8150	52
Sonokeling 7	0.057	0.120	1.231	7.410	880	1.5366	12595	18.21	1.84	2.87	7.26	11.57	6447	48
Sonokeling 8	0.059	0.120	1.232	7.535	869	1.5674	12961	24.35	2.47	3.63	9.69	13.55	7555	62
Sonokeling 9	0.062	0.121	1.231	7.740	840	1.5060	11551	24.86	2.53	3.68	9.90	13.19	7981	60
Sonokeling 10	0.058	0.117	1.233	6.530	777	1.7600	14638	20.25	2.06	3.21	8.03	10.60	8554	53
Average (Ẋ)						1.5165	11965	Average (Ẋ)					7838	56
Minimum						1.1645	6365	Minimum					6447	46
Maximum						1.7600	14638	Maximum					8554	65

E. Static and Dynamic Evaluation of Munggur Wood

Sample	**Material Data**					**Dynamic Evaluation**		**Static Evaluation**						
	h	**b**	**Lo**	**W**	**ρ**	**f**	**MOE_d**	**F_{max}**	**F_1**	**w_1**	**F_2**	**w_2**	**MOE_s**	**MOR**
	(m)	**(m)**	**(m)**	**(kg)**	**(kg/m^3)**	**(kHz)**	**(MPa)**	**(kN)**	**(kN)**	**(mm)**	**(kN)**	**(mm)**	**(MPa)**	**(MPa)**
Munggur 1	0.061	0.120	1.231	4.790	532	1.1908	4569	13.96	1.41	4.03	5.60	15.43	3806	34
Munggur 2	0.061	0.121	1.232	5.180	572	1.4360	7161	11.92	1.23	2.25	4.69	9.35	5009	29
Munggur 3	0.060	0.122	1.233	4.925	543	1.1208	4151	12.34	1.26	3.65	4.94	13.74	3717	30
Munggur 4	0.061	0.120	1.232	5.370	600	1.1208	4575	13.08	1.33	3.72	5.22	13.78	4001	32
Munggur 5	0.059	0.121	1.232	5.515	628	1.5235	8851	10.33	1.03	1.92	4.12	7.63	5566	26
Munggur 6	0.059	0.120	1.230	4.815	553	1.3178	5811	11.89	1.22	2.75	4.74	11.35	4236	30
Munggur 7	0.060	0.120	1.232	4.600	521	1.4273	6446	9.98	1.01	2.60	3.98	9.65	4372	25
Munggur 8	0.058	0.120	1.232	4.920	571	1.5893	8754	11.89	1.23	2.29	4.75	8.44	5922	31
Munggur 9	0.058	0.120	1.232	5.450	636	1.1077	4735	13.37	1.39	4.03	5.34	15.56	3538	35
Munggur 10	0.060	0.120	1.232	5.250	592	1.2083	5246	12.56	1.26	3.39	5.01	14.49	3500	31
Average ($\dot{X}$)						1.3042	6030	Average ($\dot{X}$)					4367	30
Minimum						1.1077	4151	Minimum					3500	25
Maximum						1.5893	8851	Maximum					5922	35

F. Static and Dynamic Evaluation of Sukun Wood

Sample	Material Data					Dynamic Evaluation		Static Evaluation						
	h	**b**	**Lo**	**W**	**ρ**	**f**	**MOE_d**	**F_{max}**	**F_1**	**w_1**	**F_2**	**w_2**	**MOE_s**	**MOR**
	(m)	**(m)**	**(m)**	**(kg)**	**(kg/m^3)**	**(kHz)**	**(MPa)**	**(kN)**	**(kN)**	**(mm)**	**(kN)**	**(mm)**	**(MPa)**	**(MPa)**
Sukun 1	0.06	0.1183	1.231	1.945	223	1.9030	4886	4.15	0.42	1.09	1.63	4.70	3534	11
Sukun 2	0.059	0.1177	1.229	2.14	251	1.9133	5546	6.76	0.68	1.56	2.68	6.67	4134	18
Sukun 3	0.06	0.117	1.231	4.88	565	1.2872	5671	11.01	2.21	5.86	4.40	11.39	4204	28
Sukun 4	0.06	0.1198	1.232	4.955	560	1.6111	8818	14.87	2.97	5.99	5.92	11.25	5827	37
Sukun 5	0.061	0.1183	1.231	3.965	446	1.0419	2937	6.36	0.66	1.60	2.51	6.17	4251	16
Sukun 6	0.06	0.1197	1.231	4.055	459	1.3338	4946	6.50	0.69	1.74	2.65	7.00	3845	16
Sukun 7	0.0605	0.1196	1.231	5.655	635	1.4054	7601	14.36	1.39	2.62	5.71	10.06	6027	36
Sukun 8	0.058	0.1197	1.23	5.075	594	1.6155	9386	14.70	1.48	2.7774	5.91	10.22	6174	38
Sukun 9	0.0602	0.1195	1.231	3.525	398	1.1733	3321	0.34	0.10	0.11	0.15	0.29	2779	0,86
Sukun 10	0.0605	0.12	1.233	3.54	395	1.2740	3903	2.53	0.29	0.68	0.98	3.27	2750	6
Average ($\dot{X}$)						1.4872	5966	Average ($\dot{X}$)					4527	23
Minimum						1.0419	2937	Minimum					2750	6
Maximum						1.9133	9386	Maximum					6174	38

G. Dynamic Evaluation of Steal Beam (Benchmark of LSWM)

Location : Structure Engineering Lab Civil Engineering UGM

$MOE_d = C^2 . \rho = (2. L. f)^2 . \rho$

$MOE_d = 4. L.^2 f .^2 \rho$

where:

f = frequency (kHz)

L = length of wood beam (m)

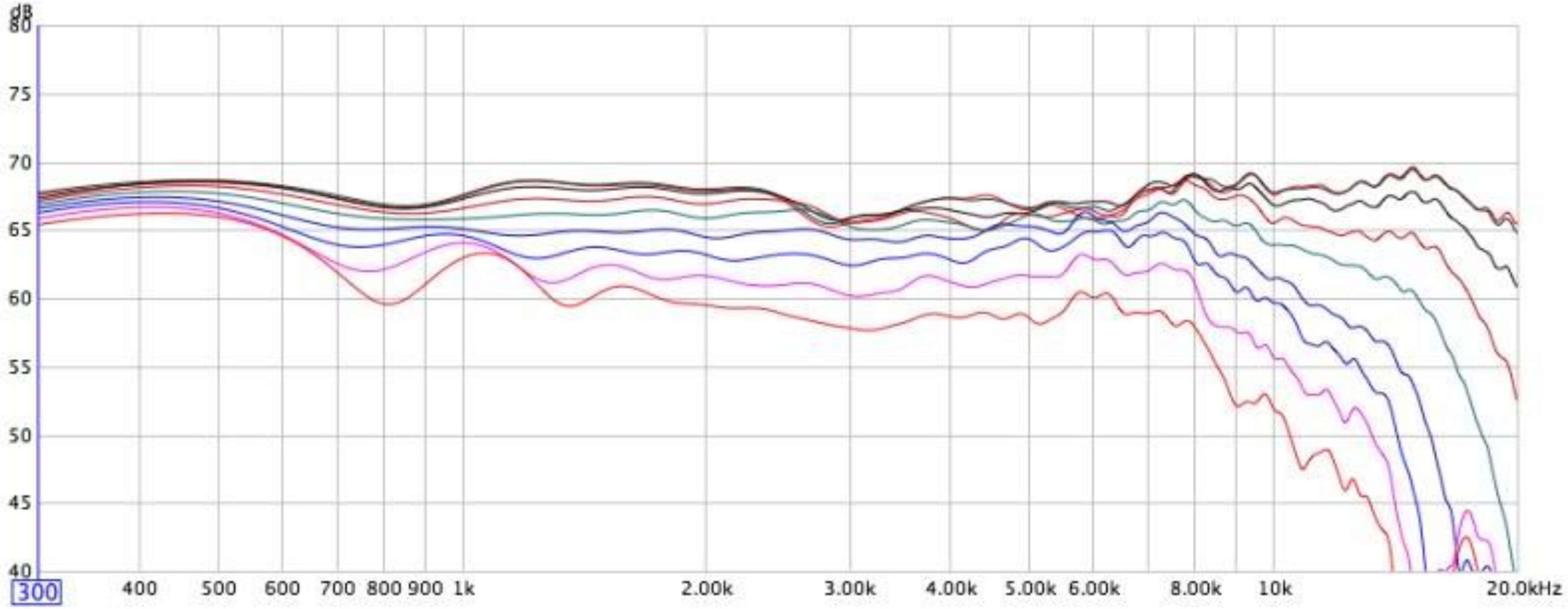
ρ = density of wood (kg/m^3)

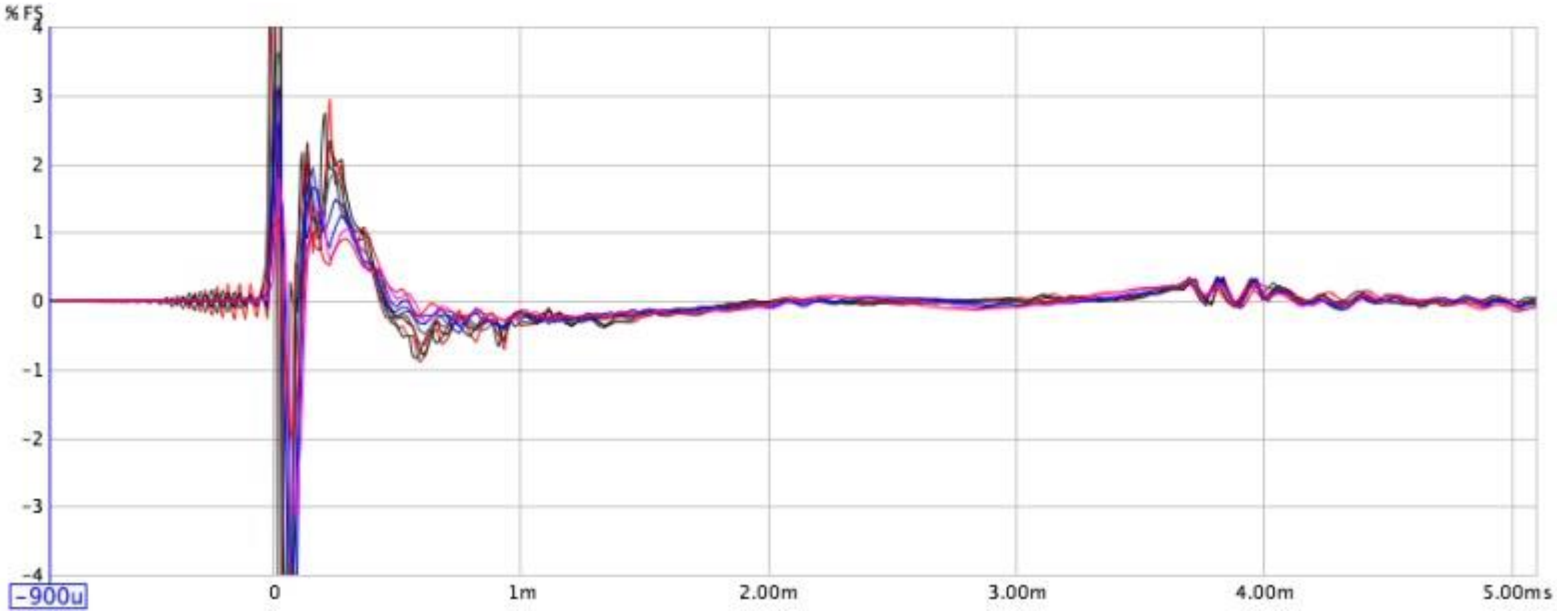
Sample	Lo (m)	Volume (m^3)	W (kg)	ρ (kg/m^3)	f (kHz)	MOE_d (MPa)
1	0.65	0.000187324	1.325	7073.325	3.9530	186794.44
2	0.65	0.000187324	1.325	7073.325	3.9708	188480.47
3	0.65	0.000187324	1.325	7073.325	3.9621	187655.45
4	0.65	0.000187324	1.325	7073.325	3.9665	188072.47
5	0.65	0.000187324	1.325	7073.325	3.9665	188072.47
Average (Ẋ)					3.9619	187637.45
Deviation Standard (*S*)					0.0126	1192.20
Percentile 5% = Ẋ- 1,645 *S*					3.9412	185676.29

H. Behringer TRUTH B1030A Frequency Response

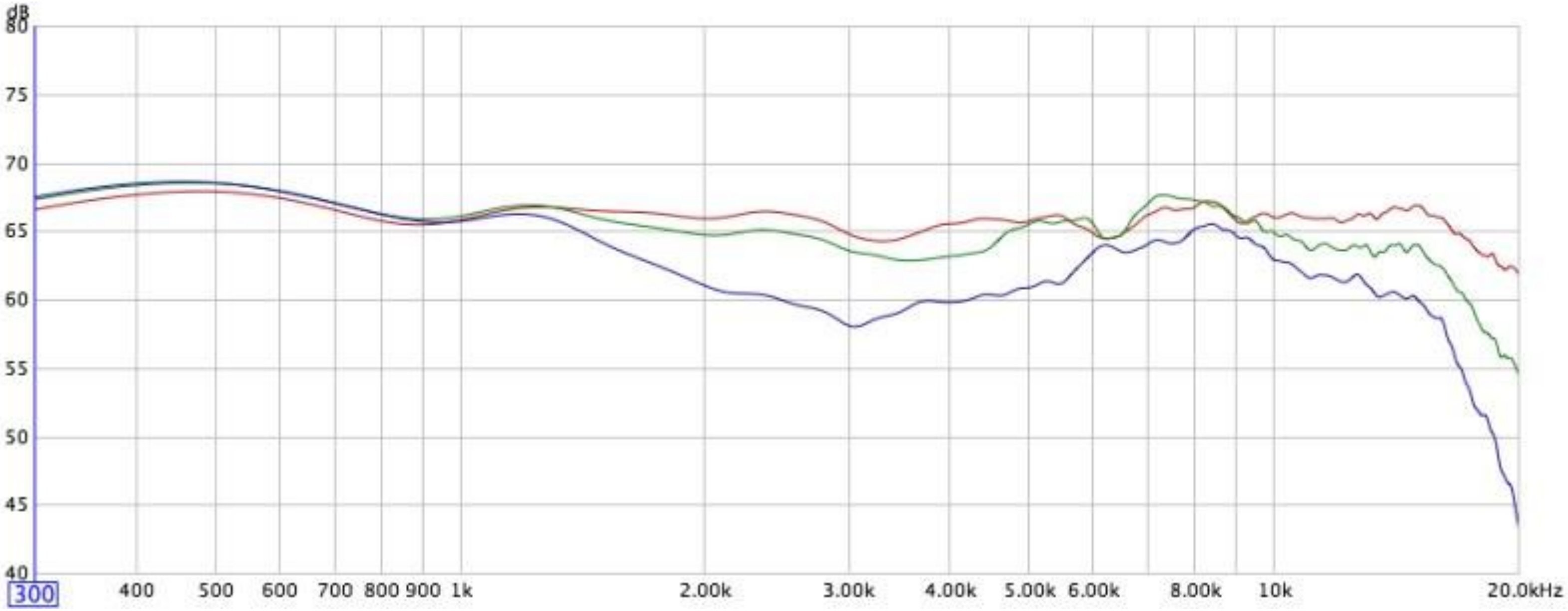
Below is some measurement of the speaker frequency response done by [24]

1) 0-90 Degrees in 11.25 Degree Steps

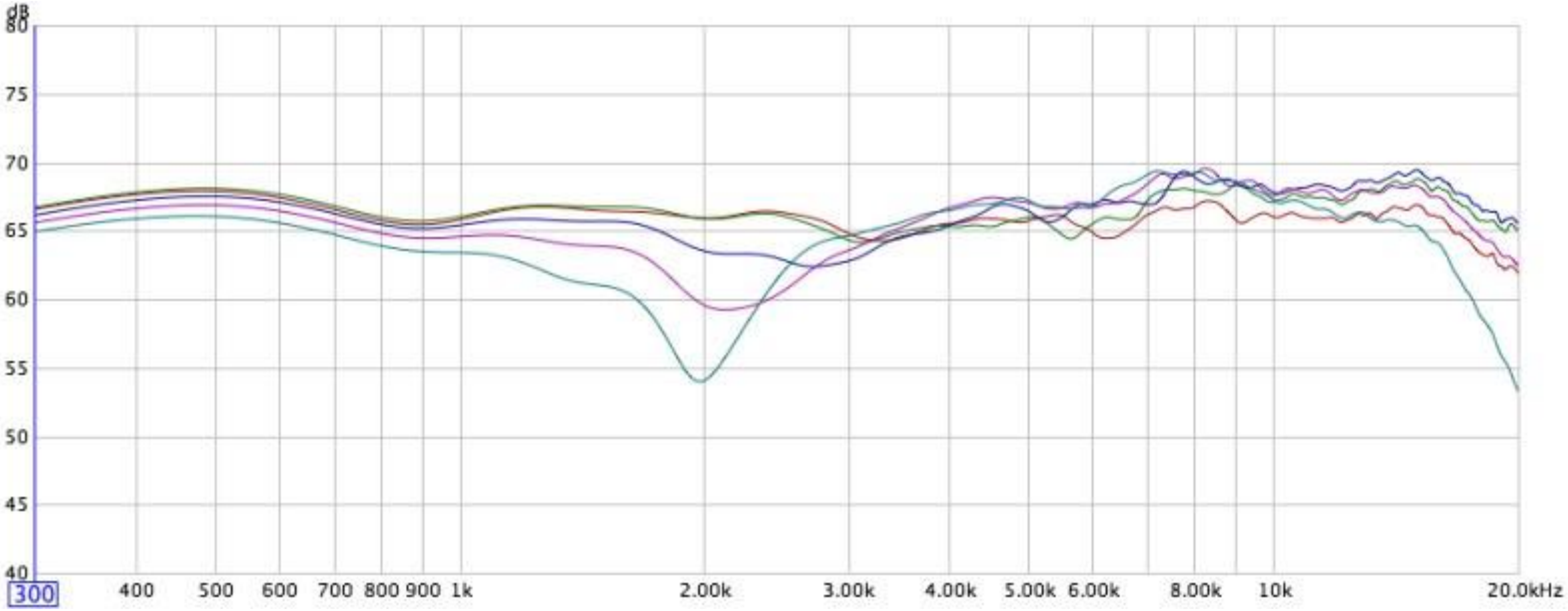


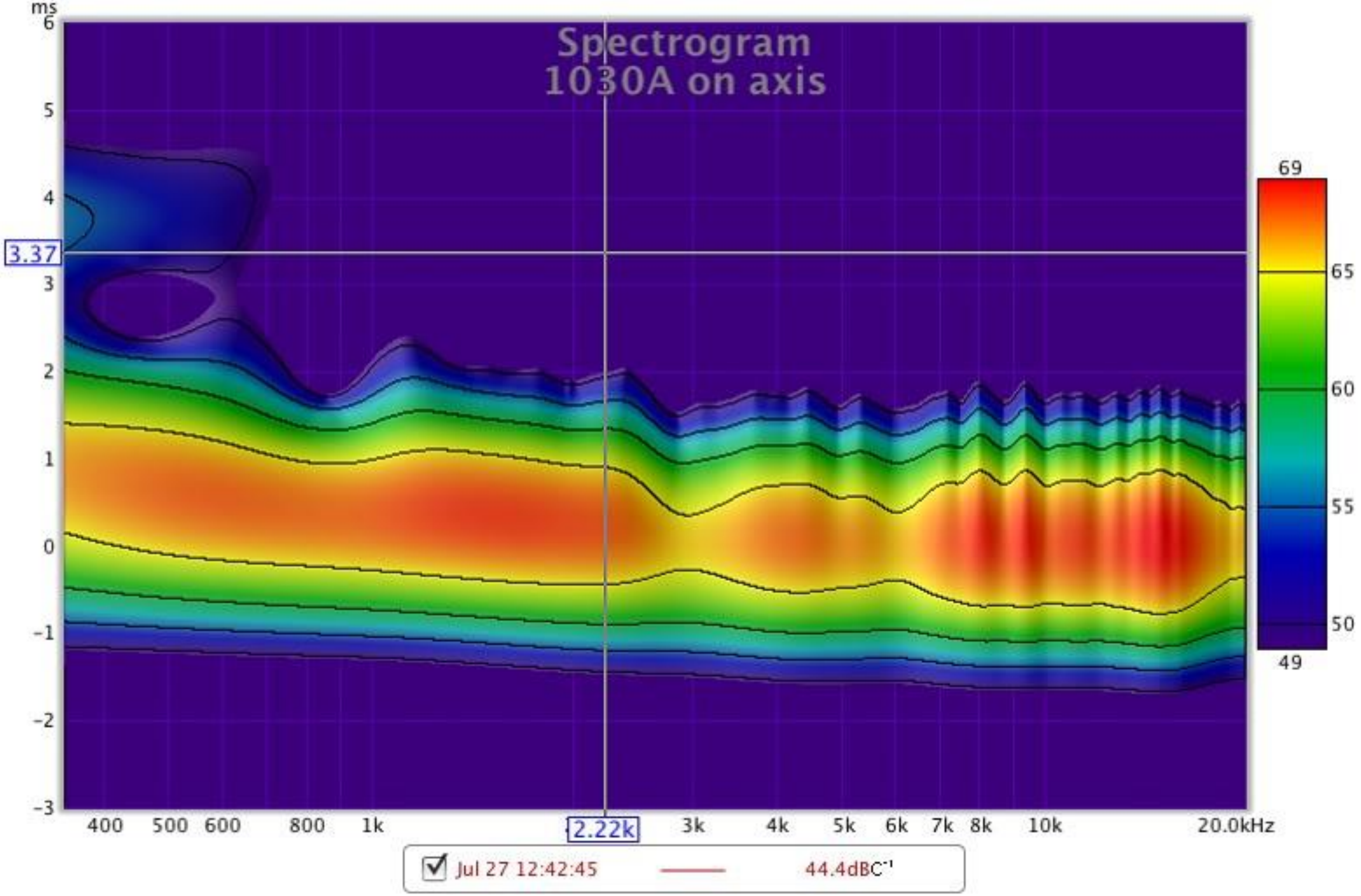

% FS
4
3
2
1
0
-1
-2
-3
-4
-900u
0
1m
2.00m
3.00m
4.00m
5.00ms

2) Toward woofer 0-22.5 degrees in 11.25 degree steps:

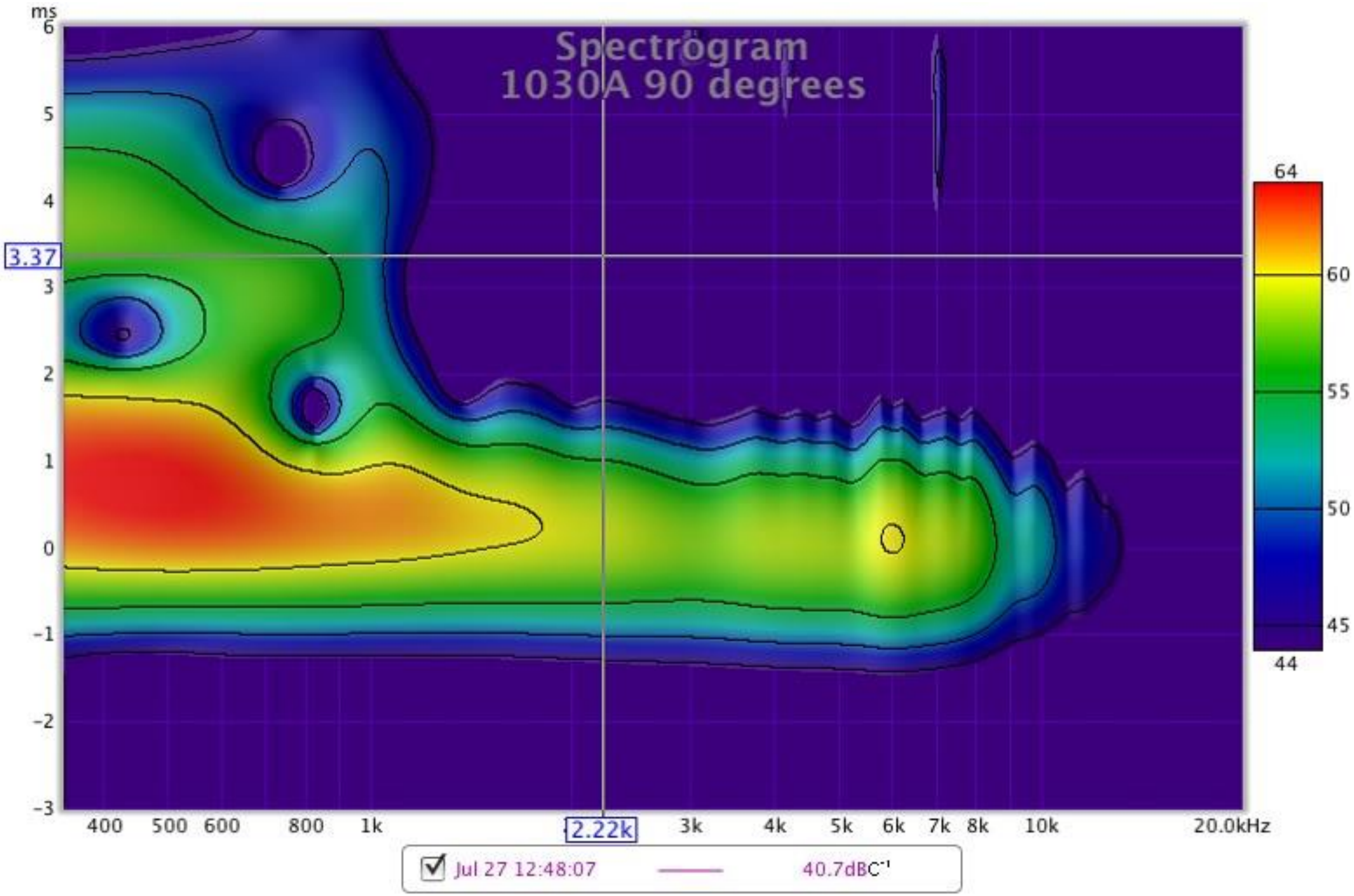


3) Toward tweeter 0-45 degrees in 11.25 degree steps:





Spectrogram
1030A on axis
ms
6
5
4
3.37
3
2
1
0
-1
-2
-3
400
500
600
800
1k
2.22k
3k
4k
5k
6k
7k
8k
10k
20.0kHz
69
65
60
55
50
49
Jul 27 12:42:45
44.4dBC



Spectrogram
1030A 90 degrees
ms
6
5
4
3.37
3
2
1
0
-1
-2
-3
400 500 600 800 1k 2.22k 3k 4k 5k 6k 7k 8k 10k 20.0kHz
64
60
55
50
45
44
Jul 27 12:48:07
40.7dBC

APPENDIX.4 Resulted Raw Data

A. Microphone Performance Test

average-freq	average-dB	average-freq-kHz	SNR	threshold	noise
808.00	-18.4	0.8	50.76	1	-69.2
904.00	-19.5	0.9	51.54	1	-71
1002.00	-21.1	1.0	50.82	1	-71.9
1100.20	-22.0	1.1	50.56	1	-72.6
1198.00	-13.1	1.2	58.08	1	-71.2
1299.00	-20.8	1.3	50.44	1	-71.2
1403.80	-20.1	1.4	52.54	1	-72.6
1509.80	-24.2	1.5	49.36	1	-73.6
1606.00	-25.5	1.6	47.74	1	-73.2
1703.00	-25.9	1.7	47.54	1	-73.4
1801.00	-24.1	1.8	50.20	1	-74.3
1899.00	-25.0	1.9	49.62	1	-74.6
1998.00	-26.0	2.0	47.60	1	-73.6
2101.20	-26.7	2.1	47.50	1	-74.2
2206.60	-27.0	2.2	47.82	1	-74.8
2307.60	-28.0	2.3	47.12	1	-75.1
2404.00	-30.3	2.4	44.90	1	-75.2
2502.00	-34.3	2.5	41.04	1	-75.3
2601.00	-38.0	2.6	37.30	1	-75.3
2698.00	-38.9	2.7	36.26	1	-75.2
2799.00	-37.7	2.8	37.72	1	-75.4
2904.00	-36.3	2.9	38.70	1	-75
3008.40	-33.6	3.0	40.00	1	-73.6
3105.80	-33.3	3.1	41.58	1	-74.9
3202.00	-33.5	3.2	42.08	1	-75.6
3302.00	-35.1	3.3	40.56	1	-75.7
3399.00	-34.1	3.4	41.60	1	-75.7
3500.00	-33.3	3.5	42.48	1	-75.8
3600.00	-32.3	3.6	43.52	1	-75.8
3711.40	-31.5	3.7	44.30	1	-75.8
3808.00	-31.3	3.8	44.00	1	-75.3
3905.00	-30.2	3.9	45.10	1	-75.3
4003.00	-28.4	4.0	46.68	1	-75.1
4102.00	-27.2	4.1	47.90	1	-75.1

B. Soundcard Performance Test

Frequency-Hz	Sine		Ramp	Square
	Gain-dB	Freq-FFT-Hz		
800	-11.7	761.00		
900	-12.1	881.00		
1000	-11.8	1001.00		
1100	-12.1	1134.00		
1200	-12.0	1205.00		
1300	-12.1	1301.00		
1400	-12.2	1409.00		
1500	-11.9	1512.00		
1600	-11.9	1607.00		
1700	-11.8	1706.00		
1800	-11.8	1804.00		
1900	-11.4	1900.00		
2000	-12.2	2001.00		
2100	-12.2	2110.00		
2200	-12.0	2210.00		
2300	-11.8	2309.00		
2400	-11.8	2405.00	Yes	Yes
2500	-11.8	2500.00		
2600	-11.8	2600.00		
2700	-11.9	2703.00		
2800	-12.0	2800.00		
2900	-12.2	2905.00		
3000	-11.9	3031.00		
3100	-11.8	3106.00		
3200	-11.8	3202.00		
3300	-11.8	3304.00		
3400	-11.9	3406.00		
3500	-11.7	3497.00		
3600	-12.0	3616.00		
3700	-12.1	3707.00		
3800	-11.4	3810.00		
3900	-11.8	3913.00		
4000	-12.1	4067.00		
4100	-11.8	4100.00		

C. Exponential Attenuation Data

1) Sonokeling

Sonokeling					
1		2		3	
h	**R^2**	**h**	**R^2**	**h**	**R^2**
6.30E-02	0.7420	8.40E-02	0.9336	1.01E-01	0.9133
1.09E-01	0.9165	1.05E-01	0.9470	1.02E-01	0.9237
8.30E-02	0.8070	6.40E-02	0.5274	8.90E-02	0.8425
9.80E-03	0.7120	9.10E-02	0.8131	1.03E-01	0.6783
7.60E-02	0.9819	7.60E-02	0.9794	8.50E-02	0.9606
7.70E-02	0.9581	7.60E-02	0.9522	7.60E-02	0.9444
8.40E-02	0.9879	8.10E-02	0.9832	8.30E-02	0.9687
6.50E-02	0.9536	7.20E-02	0.9930	6.90E-02	0.9888
1.07E-01	0.9886	9.30E-02	0.9784	9.00E-02	0.9430
6.70E-02	0.9640	6.70E-02	0.9468	6.70E-02	0.9377

2) Teak

Teak					
1		2		3	
h	**R^2**	**h**	**R^2**	**h**	**R^2**
6.80E-02	0.9919	5.90E-02	0.9931	5.80E-02	0.8374
5.20E-02	0.9036	4.40E-02	0.8192	5.20E-02	0.9459
5.70E-02	0.9694	5.70E-02	0.9851	5.70E-02	0.9537
7.20E-02	0.9785	7.90E-02	0.9873	7.55E-02	0.9829
6.00E-02	0.8410	6.80E-02	0.7187	4.70E-02	0.9277
9.40E-02	0.8840	1.14E-01	0.9220	9.60E-02	0.9280
8.10E-02	0.8290	6.40E-02	0.9319	5.50E-02	0.9652
5.30E-02	0.9845	6.50E-02	0.9685	7.10E-02	0.9789
5.30E-02	0.9653	4.50E-02	0.9846	4.00E-02	0.9270
8.10E-02	0.9615	8.40E-02	0.9861	8.60E-02	0.9838

3) Sukun

Sukun					
1		2		3	
h	R^2	h	R^2	h	R^2
8.10E-02	0.9877	3.50E-02	0.6256	2.80E-02	0.7131
4.80E-02	0.7493	6.10E-02	0.9946	6.30E-02	0.9884
6.00E-02	0.9115	6.10E-02	0.9796	6.30E-02	0.9800
4.10E-02	0.9907	4.00E-02	0.9877	4.20E-02	0.9888
4.00E-02	0.8725	4.50E-02	0.9554	5.90E-02	0.5563
4.60E-02	0.9605	5.70E-02	0.9696	5.40E-02	0.9538
3.70E-02	0.8881	4.20E-02	0.8715	3.80E-02	0.8740
5.70E-02	0.9804	4.50E-02	0.9096	3.90E-02	0.6964
9.20E-02	0.8014	7.70E-02	0.8672	7.50E-02	0.7637
1.00E-01	0.8088	1.27E-01	0.9229	9.30E-02	0.9326

4) Acacia

Acacia					
1		2		3	
h	R^2	h	R^2	h	R^2
8.10E-02	0.9452	8.80E-02	0.9652	9.50E-02	0.9528
8.60E-02	0.9665	8.90E-02	0.9624	8.40E-02	0.9603
1.07E-01	0.9798	1.10E-01	0.9930	9.70E-02	0.9927
1.05E-01	0.7925	1.04E-01	0.6464	1.04E-01	0.7453
6.10E-02	0.8230	6.30E-02	0.8787	6.80E-02	0.9175
9.70E-02	0.8188	7.70E-02	0.7982	6.40E-02	0.7407
6.30E-02	0.9936	6.40E-02	0.9906	6.50E-02	0.9904
6.40E-02	0.9781	7.30E-02	0.9850	8.80E-02	0.9856
9.60E-02	0.9560	1.01E-01	0.9315	9.90E-02	0.9476
7.30E-02	0.9935	7.00E-02	0.9825	6.90E-02	0.9933

5) Munggur

Munggur					
1		2		3	
h	R^2	h	R^2	h	R^2
7.30E-02	0.9932	7.80E-02	0.9814	7.40E-02	0.9948
7.50E-02	0.8202	1.14E-01	0.9133	9.10E-02	0.9951
6.10E-02	0.9871	6.00E-02	0.9862	6.00E-02	0.9862
6.40E-02	0.9942	6.70E-02	0.9904	6.40E-02	0.9945
1.16E-01	0.9693	1.06E-01	0.9357	1.20E-01	0.9854
6.60E-02	0.0640	6.40E-02	0.9872	8.00E-02	0.8235
8.00E-02	0.8235	1.01E-01	0.8034	9.00E-02	0.8484
8.40E-02	0.7768	7.80E-02	0.7619	8.00E-02	0.7513
6.30E-02	0.9961	6.20E-02	0.9982	6.20E-02	0.9971
8.30E-02	0.9898	8.20E-02	0.9915	8.60E-02	0.9874

6) Mahogany

Mahogany					
1		2		3	
h	R^2	h	R^2	h	R^2
5.80E-02	0.9558	5.90E-02	0.9739	6.00E-02	0.9751
7.80E-02	0.9676	8.10E-02	0.9704	7.80E-02	0.9895
7.10E-02	0.9527	6.10E-02	0.9314	6.60E-02	0.9414
5.00E-02	0.9876	6.10E-02	0.9506	5.55E-02	0.0583
6.30E-02	0.4657	6.60E-02	0.9429	6.70E-02	0.9429
5.60E-02	0.9853	6.10E-02	0.9668	5.50E-02	0.9823
6.60E-02	0.9509	8.20E-02	0.9171	6.80E-02	0.9727
7.60E-02	0.9803	6.70E-02	0.9872	6.70E-02	0.9815
6.20E-02	0.9547	6.30E-02	0.9626	5.90E-02	0.9349
5.70E-02	0.9989	5.40E-02	0.9964	5.60E-02	0.9982