UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA

FACULTATEA DE MATEMATICǍ ŞI INFORMATICǍ

SPECIALIZAREA INFORMATICĂ-ROMÂNĂ

LUCRARE DE LICENŢĂ

Detecția și eliminarea distorsiunii din înregistrări audio de pe formate analoage

Conducător ştiinţific

Lect. Dr. Sterca Adrian

Absolvent

**Drimba Alexandru**

**2018**

BABEŞ-BOLYAI UNIVERSITY CLUJ-NAPOCA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

SPECIALIZATION COMPUTER SCIENCE - ROMANIAN

DIPLOMA THESIS

Distortion detection and removal on audio recordings from analog formats

Supervisor

Lect. Dr. Sterca Adrian

Author

**Drimba Alexandru**

**2018**

Abstract

Contents

1. Introduction. Problem statement and motivation.
   1. A brief history of audio recording formats. Analog vs digital
   2. Mechanical analog storage. Recording and playback. Transcription of audio signal to digital format
   3. Causes of distortion in mechanical analog formats
   4. Purpose of this work
2. Related work (applications).
   1. Audacity
   2. Nero WaveEditor
   3. ClickRepair
3. Basic digital signal processing
   1. Digital audio signal representation
   2. Filters: Finite Impulse Response and Infinite Impulse Response
   3. Frequency domain and Fourier trasforms
   4. Filter design and frequency equalization
4. Detecting and correcting distortion
   1. Automated marking of distorted samples using neural networks
   2. Extrapolation and linear prediction
   3. Burg’s method for calculating LP coefficients
   4. Repairing the distorted sample intervals
5. Application
   1. Audio Data Sources
      1. File Storage. WAV and AU formats
      2. Caching
      3. Version control on an audio project
   2. Applying effects on data sources.
      1. FIR and IIR Filters. Equalizers.
      2. Linear prediction for sample repair
      3. Finding the distorted regions in an audio recording. Correction
      4. Signal pre- and post processing

6. Conclusions and Future Work

Chapter 3

Basic digital signal processing

3.1. Digital audio signal representation

Sound is defined as an oscillation in pressure, stress, particle displacement, particle velocity, etc., propagated in a medium with internal forces (e.g., elastic or viscous), or the superposition of such propagated oscillation [1]. This oscillation can be represented as a continuous function that describes the variation in time of the medium’s pressure, allowing us to “see” sounds. (Fig. 3.1)

Sound is transmitted through gases and liquids as longitudinal waves, and through solids both as longitudinal and transverse waves. A transmitting medium is required, so sound cannot travel through vacuum. In a longitudinal wave, the direction of displacement is the same as the direction of propagation, while in a transverse wave, the direction of displacement is perpendicular to the direction of propagation [2]. To better understand the difference between the longitudinal and transversal waves, Fig. 3.1 depicts sound waves in air (longitudinal waves), while Fig. 3.2 presents transverse waves travelling through a metal wire.

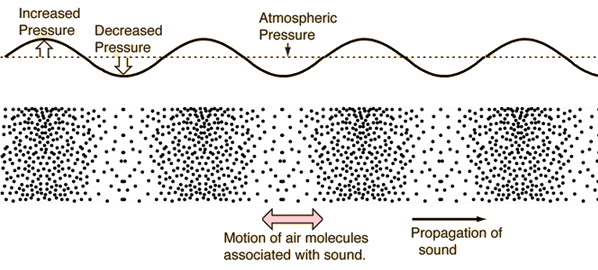
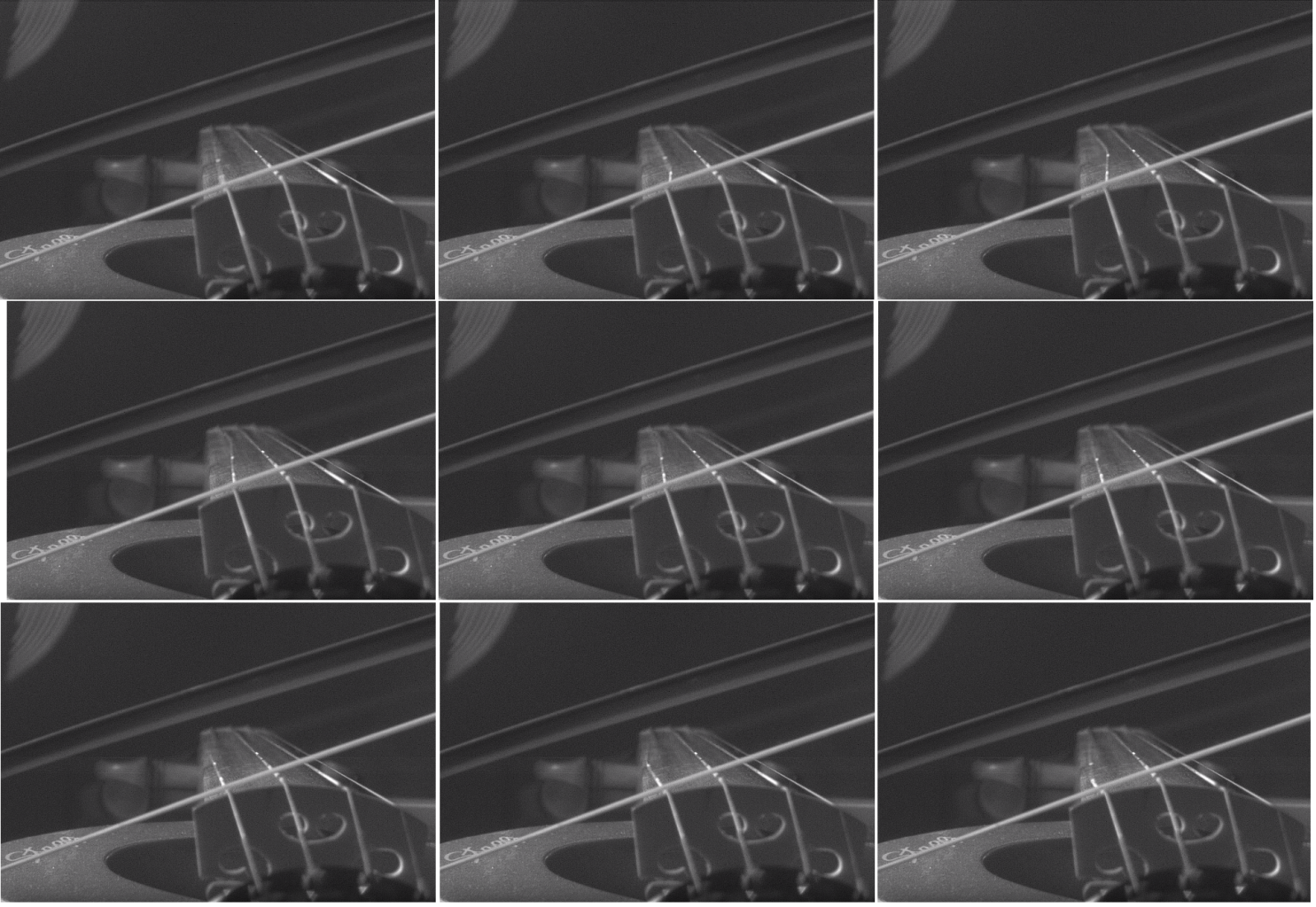


Fig. 3.1: Sound Waves in Air. A single-frequency sound wave traveling through air will cause a sinusoidal pressure variation in the air. The air motion which accompanies the passage of the sound wave will be back and forth in the direction of the propagation of the sound, a characteristic of longitudinal waves.[3]

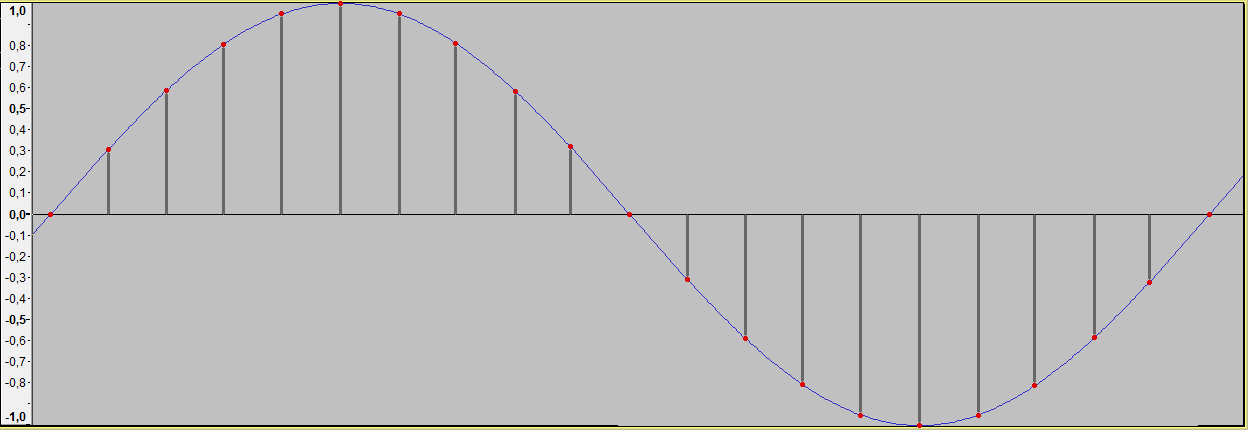
Fig. 3.2. Transverse waves seen in a bowed violin chord. [4]. The bow’s movement drags the chord, its displacements being transmitted as transverse waves moving along the chord with the speed of sound of the chord’s material.

Audio signals are representation of sound, typically as an electrical voltage (analog) or as discrete numerical values (digital). Conversion from the analog continuous-time signal to the digital discrete-time signal (also called sampling) is made usually with ADCs (Analog to Digital Converters), and with DACs(Digital to Analog Converters) from digital to analog. A **sample** isa signal’s value at a point in time**.** When converting from analog to digital, some of the information is lost because of factors like:

* 1. discretization – the resulted signal is no longer continuous (precisely defined in every point in time), but discrete: the intensity of the analog signal is recorded at fixed time points. The number of equidistant time points in a second is called **sample rate** (or sampling frequency)**.** The highest frequency that can be carried by the signal, called the **Nyquist frequency**, is given by the following formula:
  2. storage as finite numbers – as opposed to analog values, the precision of the digital values is finite, so only some of the significant digits can be stored.

By choosing an appropriate sample rate and sample encoding, the lost information can be small enough for it to be negligible.

After getting the sample values from the original analog signal, the samples are then stored as digital numbers in audio files. These sample values can be stored either as uncompressed files, like the WAV and AU formats, which we’ll be discussing about later, or as compressed files (to decrease file size). Compressed file formats can be lossy (the decompressed data is an approximation of the original), such as MP3, or lossless (compression preserves the exact original values ), such as FLAC. Audio files typically contain information about the sampling rate, number of channels and sample encoding (float/integers, signed/unsigned, bit-depth, companding and others).

Fig. 3.3. Conversion from continuous-time to discrete-time. Here, a sine wave is sampled 20 times for each cycle. Each sample is then stored as a 8-bit signed integer ( values in [ -128, 127 ] ). Table below (Table 3.1.1) shows the discretization errors raised at the conversion.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Sample no. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Actual value | 0,00000 | 0,30902 | 0,58779 | 0,80902 | 0,95106 | 1,00000 | 0,95106 | 0,80902 | 0,58779 | 0,30902 | 0,00000 |
| Sampled value | 0,00000 | 0,30469 | 0,58594 | 0,80469 | 0,94531 | 0,99219 | 0,94531 | 0,80469 | 0,58594 | 0,30469 | 0,00000 |
| Error | 0,00% | 0,43% | 0,19% | 0,43% | 0,57% | 0,78% | 0,57% | 0,43% | 0,19% | 0,43% | 0,00% |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Sample no. | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| Actual value | -0,30902 | -0,58779 | -0,80902 | -0,95106 | -1,00000 | -0,95106 | -0,80902 | -0,58779 | -0,30902 | 0,00000 |
| Sampled value | -0,31250 | -0,59375 | -0,81250 | -0,95313 | -1,00000 | -0,95313 | -0,81250 | -0,59375 | -0,31250 | 0,00000 |
| Error | 0,35% | 0,60% | 0,35% | 0,21% | 0,00% | 0,21% | 0,35% | 0,60% | 0,35% | 0,00% |

Table 3.1. Errors at conversion from analog to digital samples. The digital samples are stored as 8-bit signed integers, with values in [ -128, 127 ], rescaled here to [ -1,1 ) to show the error. Analog samples are in the range [ -1,1 ], where -1 is the smallest possible signal value, and 1 is the maximum. We can see the errors are pretty large for the chosen sample encoding.

3.2. Filters: Finite Impulse Response and Infinite Impulse Response

Signal filtering is one of the main applications of signal processing [[5], p.185]. Filters are used for many purposes, but, in audio signal processing, they are mostly used to achieve a desired frequency response, the most basic being low-pass (which only let low frequencies pass), high-pass (same for high frequencies), band-pass and band-stop filters. Low- and high-pass filters are widely used in speaker cabinets to separate the input signal to each of the drivers, depending on drivers’ frequency response. An example for use of band-pass filters is in radio communication, to isolate the required frequency band (representing a radio channel) from the others.

Depending on the considered audio signal, there are two types of filters: analog (electronic) and digital. Electronic filters can range from simple circuits, made just from simple passive components: resistors, inductors and capacitors, to complex ones, including along the usual passive components, active components: transistors, amplifiers, operational amplifiers and others.

The digital signal processing systems use samples of input signals, which constitute series of numbers. The result may be also series of numbers, to be used as output signals [[5], p. 239]. Each sample of the output signal is computed as a weighted sum of the previous few input and output samples. This weighted sum is also known as a convolution. There are two primary types of digital filters: FIR (Finite Impulse Response) and IIR (Infinite Impulse Response). The impulse response of a system is its output when presented with a brief input signal, called an impulse. Applying a filter to a signal will alter each frequency’s magnitude and phase according to the filter’s impulse response. It remains to design a filter, knowing the desired response, which we’ll discuss in the next section.

When using a FIR filter, each output sample is computed from the previous input samples and a fixed set of coefficients, the weights. The weights are associated to input samples based on each sample’s delay (how far it is from the most recent input sample). Input samples, ordered by delay, are stored in a so-called delay line. The resulted sample is calculated by a set of multiply-accumulate operations: each is input sample is multiplied with its weight, and summed to the output sample:

(3.1)

where:

• is the input signal,

• is the output signal,

• is the filter order and the number of taps. A “tap” is simply a coefficient/delay pair [6]; an th-order filter needs N previous input samples and has terms on the right-hand side,

• is the set of the coefficients; is the weight associated to the th input sample.

For an FIR filter, the filter coefficients are, by definition, the impulse response of the filter [6]. The impulse response is finite ( when the input samples become 0, the output will eventually become 0 ) because there is no feedback in the FIR. A lack of feedback guarantees that the impulse response will be finite [7].

The other class of digital filters is IIR filters which, unlike FIR filters, use feedback, i.e. samples already computed by the filter are used in the next iterations. Each output sample is computed from the previous input sample (feedforward), but also from the previous output samples (feedback). Like FIR filters, each sample has an associated weight, based on the sample’s delay, so there are needed two sets of coefficients: feedforward and feedback. The formula is as following:

(3.2)

In many digital signal processing applications, FIR filters are preferred over their IIR counterparts. The main advantages of the FIR filter designs over their IIR equivalents are the following [8]:

1. FIR filters with linear phase response (all frequencies are delayed by the same constant time amount) can easily be designed;
2. They are simple to implement (two nested loops, one for iterating the input samples, one for the coefficients);
3. FIR filters are always stable, i.e. given a bounded input signal, the output signal will also be bounded. If designed wrong, IIR filters can diverge.
4. They have desirable numeric properties. The use of finite-precision arithmetic in IIR filters can cause significant problems due to the use of feedback, but FIR filters can usually be implemented using fewer bits, and the designer has fewer practical problems to solve related to non-ideal arithmetic [7];
5. Excellent design methods are available for various kinds of FIR filters with arbitrary specification [8].

The main disadvantage of FIR filters is that they may require much more computational effort and memory than a comparable IIR counterpart.

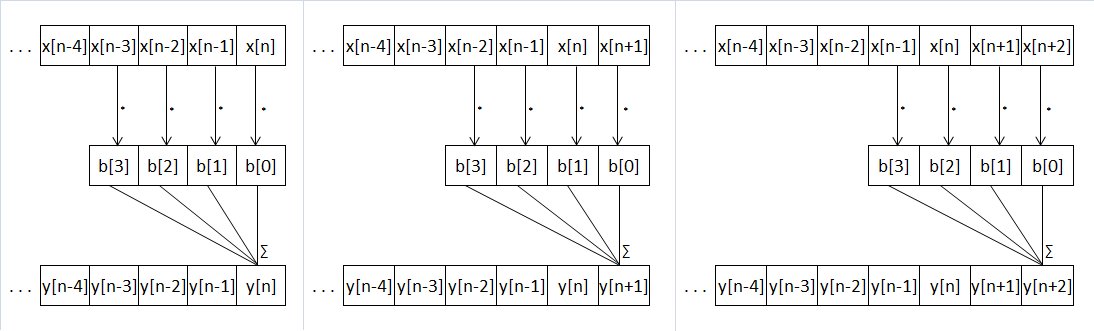
Figures 3.4 and 3.5 give an example as how an FIR filter and IIR filter work along the input data:

Fig. 3.4: Application of a digital FIR filter

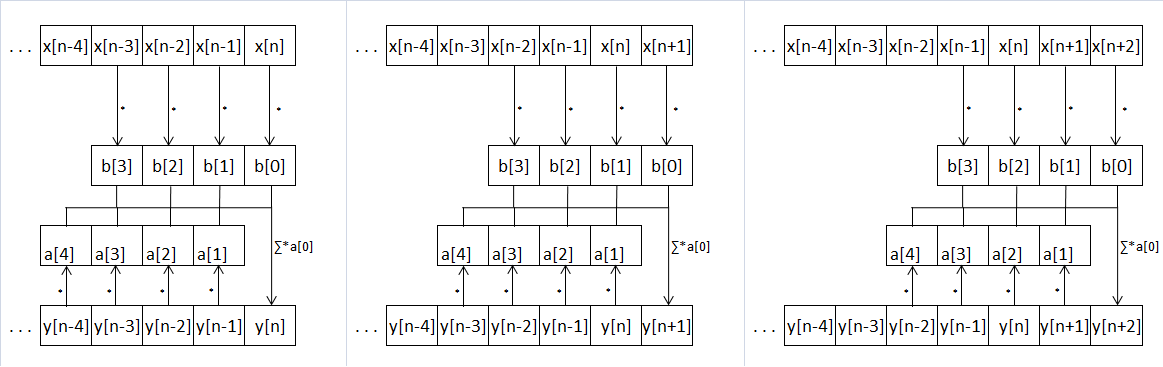


Fig. 3.5: Application of a digital IIR filter

3.3. Frequency domain and Fourier transforms

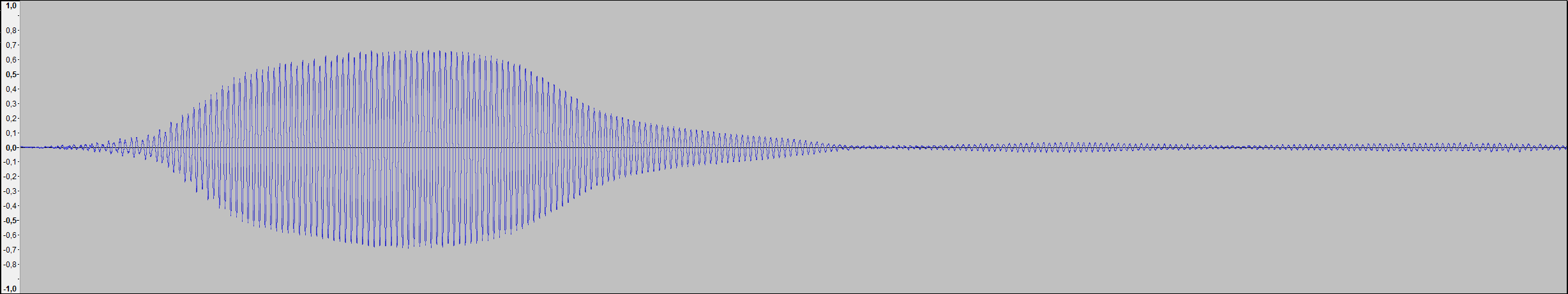
As we previously saw, an audio signal is represented as a variation of intensity over time (being it air pressure or electric voltage). The human auditory system picks up the eardrum’s vibrations and transduces them into nerve impulses, which are then perceived in the brain as “sound”. However, the brain doesn’t interpret the sound by its pressure wave, but rather by its frequencies’ amplitudes, phase and pitch. For example, an audible sine wave will be perceived not as the periodic function the sin function looks like, but as a pure tone, with a certain pitch and loudness.

The function that gives the variation of wave’s intensity is called **time-domain.** The function that gives the component frequencies is called **frequency-domain.**

In 1822, Fourier in his work on heat flow made a remarkable assertion that every function f(x) with period 2π can be represented by a trigonometric infinite series of the form:

(3.3)

[[9], p.1]

An infinite series of this form is called a Fourier series [[9], p.1]. an and bn are called the Fourier coefficients. Using the Fourier series, one can transform from time-domain to frequency domain and vice-versa. Fig. 3.6 shows the same sound, in time-domain and in frequency-domain.

background noise

fundamental (1st harmonic) – 880 Hz

2nd harmonic – 1760 Hz

3rd harmonic – 2640 Hz

4th harmonic – 3520 Hz

5th harmonic – 4400 Hz

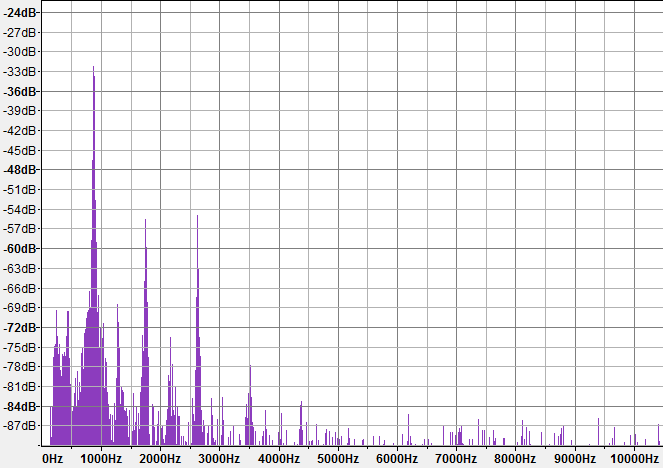
Fig. 3.6.: Note A5 (880Hz) played on a flute. Top – plot of the time-domain function: x-axis is time, y-axis is signal’s amplitude. Bottom – plot of the frequency-domain function, also called spectrogram: x-axis is time (the signal was partitioned into chunks, and the transform was made on each of those chunks), y-axis is frequency, and z-axis (color intensity) is amplitude. A note played on flute has the fundamental considerably louder than the harmonics, so the sound wave generated look pretty close to a sine wave. The spectrogram clearly shows the harmonics.

As we’ll do digital signal processing, we’ll not be working with continuous functions, but rather with discrete ones. A DFT (Discrete Fourier Transform) is a Fourier that transforms a discrete number of samples of a time wave and converts them into a frequency spectrum (Fig. 3.7). The IDFT (Inverse Discrete Fourier Transform) transforms the frequency spectrum to a discrete number of samples. The equation for the Discrete Fourier Transform is:

where F(n) is the amplitude at the frequency n, and N is the number of discrete samples taken.

Note that:

i.e. each frequency has a cosine and a sine component, each with its own amplitude.

Fig. 3.7. Spectral plot of the same signal from Fig. 3.6. This time, the whole signal was put into the Fourier transform. Harmonics can be clearly seen.

The IDFT formula looks similar to the DFT: (3.8)

3.4. Filter design and frequency equalization

We’ll want to use filters in order to

Bibliography

[1]. American National Standard on Acoustical Terminology, ANSI/ASA S1.1-2013

[2]. Definitions of longitudinal and transverse waves, <http://www.dictionary.com>

[3]. Sound waves in air, <http://hyperphysics.phy-astr.gsu.edu/hbase/Sound/tralon.html>

[4]. Bowed violin string in slow motion, <https://www.youtube.com/watch?v=6JeyiM0YNo4> (11.04.2018)

[9]. Tang, Kwong-Tin: Mathematical Methods for Engineers and Scientists 3. Fourier Analysis, Partial Differential Equations and Variational Methods, 2017

[]. A tutorial on Burg's method, algorithm and recursion, Cedrick Collomb, 2009:

<http://www.emptyloop.com/technotes/A%20tutorial%20on%20Burg's%20method,%20algorithm%20and%20recursion.pdf>

[10]. Discrete and Fast Fourier Transforms: <http://www.alwayslearn.com/DFT%20and%20FFT%20Tutorial/DFTandFFT_BasicIdea.html>

[8]. Finite Impulse Response Filter Design:

<https://www.cs.tut.fi/~ts/Mitra_Kaiser.pdf>

[5]. Jose Maria Giron-Sierra: *Digital Signal Processing with Matlab Examples, Volume I,* Springer, ISBN 978-981-10-2534-1

[]. Jose Maria Giron-Sierra: *Digital Signal Processing with Matlab Examples, Volume II,* Springer, ISBN 978-981-10-2537-2

[]. Jose Maria Giron-Sierra: *Digital Signal Processing with Matlab Examples, Volume III,* Springer, ISBN 978-981-10-2540-2

[6]. FIR Filter Design: <http://shodhganga.inflibnet.ac.in/bitstream/10603/24055/10/10_chapter%205.pdf>

[7]. FIR Filter Basics: <http://dspguru.com/dsp/faqs/fir/basics/>

[]. FIR Filter Properties: <http://dspguru.com/dsp/faqs/fir/properties/>

[]. FIR Filter Design: <https://dspguru.com/dsp/faqs/fir/design/>

[]. IIR Filter Basics: <http://dspguru.com/dsp/faqs/iir/basics/>