Insert here your thesis' task.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Exploring use of non-negative matrix factorization for lossy audio compression

Bc. Tomáš Drbota

Department of Theoretical Computer Science
Supervisor: doc. Ing. Ivan Šimeček, Ph.D.

March 10, 2019

Acknowledgements

TODO

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Article 46(6) of the Act, I hereby grant a nonexclusive authorization (license) to utilize this thesis, including any and all computer programs incorporated therein or attached thereto and all corresponding documentation (hereinafter collectively referred to as the "Work"), to any and all persons that wish to utilize the Work. Such persons are entitled to use the Work in any way (including for-profit purposes) that does not detract from its value. This authorization is not limited in terms of time, location and quantity.

In Prague on March 10, 2019 .....

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Drbota, Tomáš. *Exploring use of non-negative matrix factorization for lossy audio compression.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

Abstrakt

TODO

Klíčová slova TODO

Abstract

Non-negative matrix factorization has been successfully applied in various scenarios, mostly for analyzing large chunks of data and finding patterns in them for later use. Due to the nature of NMF, it has also seen some use in the field of image compression.

The purpose of this thesis is to research possible uses of non-negative matrix factorization in the problem of audio compression. A reference audio encoder and decoder using NMF will be implemented and various experiments using this encoder will be conducted. The results will be measured and compared to existing audio compressing solutions.

Keywords lossy, audio, compression, processing, nmf, encoding

Contents

List of Figures

Introduction

In today's age of smartphones and other portable electronic devices capable of connecting to the internet, nearly everyone has access to this giant (and still growing) library of various media, including music and other audio. However, to transmit or store all of this data in its raw uncompressed form, a large amount of bandwidth and storage would be required.

.. need for compression ..
.. common methods of audio compression ..
.. mp3 opus ..
.. this work tries nmf ..
.. state of art ..
.. then design and implement ..
.. measure results ..

Part I

Background

Chapter 1

Digital audio

Sound as we know it can be defined as a physical wave travelling through air or another means. [1] It can be measured as change in air pressure surrounding an object. Once we have this electrical representation of the wave, we can convert it back and consequently play using speakers.

In the real world, these sound waves are generally composed of many different kinds of waves, with differing frequencies and amplitudes. The human ear can tell the difference between high (whistling) and low frequencies (drums), and knowledge of this will be useful later when we are discussing audio encoding.

- TODO image of audio signal -

1.1 Important terms

.. TODO .. sampling nyquist frequency/limit quantization windowing

1.2 Digital audio representation

Most commonly, the amount of air pressure is sampled many times a second and after being processed this information is stored as a discrete-time signal using numerical representations - this is what's known as a *digital audio signal*. This entire process is called *digital audio encoding*.

By sampling the audio signal, we will potentially be losing out on some information, but given a high enough sampling rate, the result will be imperceptible to the human ear. For general purpose audio and music, the standard sampling rate is 48 kHz, alternatively 44.1 kHz from the compact disk era.

Once we have our digital signal, there are two distinct kinds of ways we can represent it. Both of them have many different data models for encoding [1], but in this work I am only going to focus on the most relevant ones.

1.2.1 Time domain representation

In the time domain, the signal is simply represented as a function of time, where t is the time and $x(t)$ is the raw amplitude, or air pressure, at that point. [2]

This is the most straightforward representation since it directly correlates to how the signal is being captured in the first place. However, as we will see later, this format is not ideal for storing audio data with any sort of compression.

1.2.1.1 PCM

In the time domain, the most basic encoding we can use is PCM (Pulse Code Modulation). After sampling a signal at uniform intervals, the discrete values are quantized; that is, each range of values is assigned a symbol in (usually) binary code.

For example using 16-bit signed PCM, each sample will be represented as a 16-bit signed integer, or in the case of multiple channels, N 16-bit signed integers, where N is the amount of channels.

PCM serves as a good base for what we are going to talk about next - Frequency domain representation and encoding.

1.2.2 Frequency domain representation

While it's simple to understand and work with for the computer with samples in the form of a sequence of amplitudes, it's difficult to run any sort of meaningful analysis on such data. To better grasp the structure of the audio we're working with, it would be helpful to be able to decompose it into its basic building blocks, so to speak. And that's where frequency based representation comes in.

The goal here is to represent the signal as not a function of time, but rather a function of frequency $X(f)$. That is, instead of having a simple sequence of amplitudes, we will have information about the magnitude for each component from a set of frequency ranges. This description alone is generally more compact than the PCM representation [2] on top of providing us with useful information about the signal, so it will serve as a good entry point to our compression schemes.

1.2.2.1 Fourier transform

Fourier transform is the first and arguably the most used tool for converting a signal from a function of time $x(t)$ into a function of frequency $X(f)$.

It is based on the *Fourier series*, which is essentially a representation of a periodic function as the linear combination of sines and cosines. [3] However, the main difference is that our function need not be periodic.

The Fourier transform of a continuous signal s is defined as: [4]

$$S(\xi) = \int_{-\infty}^{\infty} s(t) e^{-2\pi i t \xi} dt$$

The output is a complex number, which provides us with the means to find the magnitude and phase offset for the sinusoid of each frequency ξ.

The Fourier transform can also be inverted, providing us with an easy way to obtain the original signal back from its frequency components. The inverse transform is defined as:

$$s(t) = \int_{-\infty}^{\infty} S(\xi) e^{2\pi i t \xi} d\xi$$

However, seeing as our samples are discretely sampled, we will need to modify our transform accordingly.

The discrete Fourier transform of a discrete signal $s_0, s_1, ..., s_{N-1}$ is: [4]

$$S_k = \sum_{n=0}^{N-1} s_n e^{-2\pi i k n / N}$$

And our inverse is:

$$s_n = \frac{1}{N} \sum_{k=0}^{N-1} S_k e^{2\pi i k n / N}$$

The issue is, due to the nature of this process, if we run the Fourier transform on our whole signal, we will only be able to analyse it as a whole, e.g. we won't be able to tell which parts of for example a song are quiet or if there are any parts with very high frequencies - we lose our temporal data.

To alleviate this problem, we can run Fourier transform on smaller chunks of the signal, analyse them separately and later join them back into the original signal. That is the essence of the Short-time Fourier transform.

1.2.2.2 Short-time Fourier transform

1.2.2.3 Modified discrete cosine transform

.. how does sound work ..
- .. how is sound converted to digital ..
- .. basic representations (pcm) ..
- .. time-frequency representation ..
- .. state of art audio compression ..
- .. what compromises are taken ..

Chapter 2

Non-negative matrix factorization

.. what is nmf ..
- .. how is nmf defined ..
- .. what is nmf used for ..
- .. why is nmf non-negative ..
- .. different kinds of nmf ..
- .. use in audio ..

Part II

Audio compression using NMF

Chapter 3

Design

.. time domain compression ..
compressing raw
.. frequency domain compression ..
- compressing STFT
STFT
NMF
mu-law quantization, non-uniform, formula in p128 wrong (missing sgn)
- compressing MDCT
unusable, needs to be compressed consistently / losslessly
.. file structure ..
diagram for both time domain and frequency domain compression

Chapter 4

Implementation

4.1 Encoder

.. process of encoding ..
.. application of NMF ..
.. variables ..

4.2 Decoder

Chapter 5

Evaluation

.. how is audio evaluated ..
- .. gstpeaq ..
- .. how does gstpeaq work ..
- .. how and what did I test ..
- .. comparison to other formats ..

Conclusion

Bibliography

[1] You, Y. *Audio Coding Theory and Applications*. Springer US, 2010.

[2] Bosi, M.; Goldberg, R. E. *Introduction to digital audio coding and standards*. Kluwer Academic Publishers, 2003.

[3] Shatkay, H. The Fourier Transform - A Primer. Technical report, Providence, RI, USA, 1995.

[4] Recoskie, D. Constrained Nonnegative Matrix Factorization with Applications to Music Transcription. 2014.

Appendix A

Acronyms

todo TODO

Appendix B

Contents of enclosed CD

- `readme.txt` ..... the file with CD contents description
- `exe` ..... the directory with executables
- `src` ..... the directory of source codes
  - `wbdcm` ..... implementation sources
  - `thesis` ..... the directory of LaTeX source codes of the thesis
- `text` ..... the thesis text directory
  - `thesis.pdf` ..... the thesis text in PDF format
  - `thesis.ps` ..... the thesis text in PS format