pysoundanalyser

Release 0.3.4

Samuele Carcagno

Jun 09, 2023

CONTENTS:

1 What is pysoundanalyser? 1

2 Installation 3

3 User Interface 5

4 Indices and tables 7

CHAPTER

ONE

WHAT IS PYSOUNDANALYSER?

`pysoundanalyser` is an application which provides a graphical user interface to analyse short (in the range of seconds) wav files. I developed and use it mainly to quickly load and visualize the waveforms and the spectral content of sounds generated for psychoacoustics research. `pysoundanalyser` can also generate some types of sounds used in psychoacoustics research (e.g. pure tones, amplitude modulated tones, frequency modulated tones, different colors of noise, etc…).

The repository of `pysoundanalyser` is hosted on GitHub. If you find bugs, please report them there.

CHAPTER

TWO

INSTALLATION

pysoundanalyser is written in Python and requires the installation of a Python interpreter. Once the Python interpreter has been installed, pysoundanalyser can be installed via pip:

```
pip install pysoundanalyser
```

once pysoundanalyser is installed you can launch it from a bash/DOS terminal with the command

```
pysoundanalyser
```

Note that the program needs to be launched in the same Python environment in which it has been installed. The program has been tested on Linux and Windows. It should work also on Mac computers but this has not been tested. Depending on your Python distribution you may want to install the python modules pysoundanalyser depends on before installing pysoundanalyser (e.g. via the conda package manager if you're using the Anaconda Python distribution). The dependencies are:

- PyQt5
- numpy
- scipy
- pandas
- matplotlib
- PyAudio

if you're using Linux you can also install *pyalsaaudio* to have an addition sound output option. If you're using conda on Windows I'd recommend installing PyAudio via pip because the PyAudio version available on conda is not built with support for the WASAPI output interface (at least that was the case the last time I checked).

CHAPTER

THREE

USER INTERFACE

- **Load Sound** The `Load Sound` button allows you to load a wav file into the program. Currently only 16 and 32 bit wav files with one or two channels are supported. Note that the entire wav file is loaded in memory, this is fine and fast for wav files of short durations (tens of seconds), but longer sound files are going to consume huge amounts of RAM and may even halt your computer. If you need to work on long sound files, please use other software, like audacity.
  - **Save As** The `Save As` button allows you to save PSA sound objects as wav files. Currently it is only possible to save sounds as 16 or 32 bit wav files with one or two channels. If you choose a mono (1-channel) format, and multiple PSA sound objects have been selected for saving, they will be summed together before saving. If you choose a stereo (2-channels) format, "right" and "left" PSA sound objects will be saved to their respective channels in the wav file; if multiple "right" or "left" PSA sound objects have been selected, they will be summed before saving.
- **Clone Sound**
- **Concatenate**
- **Cut** The `Cut` button allows you to remove segments of a sound waveform. The starting and ending points of the segments to be cut off can be specified in seconds, milliseconds, or sample numbers. When working with sample numbers, note that PSA indexing works exactly like numpy indexing. Examples:

```
>>> import numpy as np #import numpy
>>> sig = np.arange(10) #generate a 10-elements array
>>> x[0:5] #Select the first five samples, indexing starts from 0
array([0, 1, 2, 3, 4])
>>> sig[7:10] #select last 3 samples
array([7, 8, 9])
>>> sig[1:3] #select the second and third sample
array([1, 2])
>>> sig[1:2] #select the second sample
array([1])
>>> sig[1:1] #note that if the start and end point are the same nothing is selected
array([], dtype=int64)
```

- **Play** The `Play` button allows playback of the currently selected sound.
- **Plot Waveform** The `Plot Wavform` button allows you to plot the waveform of the currently selected sound.
- **Spectrum** Plot the spectrum of the currently selected sound.
- **Spectrogram** Plot the spectrogram of the currently selected sound.
- **Autocorrelation** Plot the autocorrelation function of the currently selected sound.
- **Autocorrelogram** Plot the autocorrelogram of the currently selected sound.

- **Level Difference** Show the difference in level between two selected sounds.
- **Scale** Change the level of the currently selected sound.
- **Resample** Resample the currently selected sound.
- **Rename** Rename the currently selected sound.
- **Remove** Remove the currently selected sound from the workspace.
- **Remove All** Remove all sounds from the workspace.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search